UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA QUÍMICA
PROGRAMA DE RECURSOS HUMANOS DA AGÊNCIA NACIONAL DO PETRÓLEO

Sistema Integrado de Monitoramento de Instrumentos em Rede Foundation Fieldbus para Melhoria dos Processos de Medição e Controle na Indústria do Petróleo

Jeferson Ribeiro dos Santos Júnior - GRA
Orientador(a): ***Jorge Dantas de Melo***
Co-Orientador(a): ***Adrião Duarte Dória Neto***

Março/2008

UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
PROGRAMA DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

Sistema Integrado de Monitoramento de Instrumentos em Rede *Foundation Fieldbus* para Melhoria dos Processos de Medição e Controle na Indústria do Petróleo

Jeferson Ribeiro dos Santos Júnior

Orientador: Prof. Dr. Jorge Dantas de Melo

Co-orientador: Prof. Dr. Adrião Duarte Dória Neto

Relatório de Estágio Supervisionado apresentado ao Programa de Graduação em Engenharia de Computação da UFRN (área de concentração: Automação e Sistemas) como parte dos requisitos para obtenção do título de Engenheiro de Computação.

Natal/RN, 14 dezembro de 2007

Resumo

O presente trabalho tem o objetivo de apresentar o projeto e a implementação de um módulo adicionado ao sistema supervisório da planta LAMP, o qual permitirá o armazenamento, análise e geração de relatórios das variáveis monitoradas, no qual serão utilizadas em projetos do próprio laboratório, cuja comunicação tem como base o protocolo *Foundation Fieldbus* . São descritos os procedimentos utilizados para o desenvolvimento de aplicações industriais em tempo real na área de monitoramento de processos. Toda aplicação foi validada a partir da execução de testes reais e observação do funcionamento da aplicação, sendo analisado seu desempenho tanto para o problema de armazenamento de dados quanto para o de comunicação.

Sumário

Sumário i

Lista de Figuras iii

Lista de Tabelas iv

1 Introdução 1

1.1 Sistemas de Automação Industrial 1

1.1.1 Breve Retrospecto 2

1.1.2 Níveis de Hierarquia em Sistemas de Automação 3

1.2 Objetivos e Localização do Trabalho 4

1.3 Estrutura do Trabalho 5

2 Fundamentação Teórica 6

2.1 Comunicação entre Processos Industriais – Protocolo *Foundation Fieldbus* 6

2.1.1 Características da Tecnologia 6

2.1.2 Camada Física 7

2.1.3 Camada de Comunicação 8

2.1.4 Camada do Usuário 8

2.2 *OLE For Process Control - OPC* 9

2.2.1 Histórico 10

2.2.2 Interfaces de Acesso ao Servidor 11

2.2.3 Aspectos Práticos para Utilização do Padrão OPC 12

2.3 Sistemas Supervisórios 16

2.3.1 Componentes Físicos de um Sistema de Supervisão 17

2.3.2 Componentes Lógicos de um Sistema SCADA 18

2.3.3 Modos de Comunicação 19

2.3.4 Sistema Supervisório Elipse SCADA 19

2.4 Sistema Gerenciador de Base de Dados Relacional – PostgreSQL 22

3 Proposta de Implementação **24**
3.1 Problemática 24
3.1.1 Estrutura Disponível 24
3.2 Projeto e Implementação do Banco de Dados 25
3.3 Criação do Sistema Supervisório 27
3.3.1 Telas 28
3.3.2 Funcionamento em *Background* 30
3.3.3 Comunicação Supervisório X Banco de Dados 34
3.4 Validação do Sistema 36
3.4.1 Configuração de Rede *Foundation Fieldbus* 36
3.4.2 Comunicação rede *Foundation Fieldbus* X Supervisório 38
3.4.3 Dados Inseridos no Banco 41

4 Conclusões **43**

5 Cronograma de Execução **45**

Referências bibliográficas **46**

Lista de Figuras

1.1 Pirâmide hierárquica dos sistemas de automação 3

2.1 Modelo OSI X *Foundation Fieldbus* 7
2.2 Sistema de supervisão e controle 18

3.1 Ligação entre a rede e os instrumentos do Laboratório A 25
3.2 Estrutura projetada do Banco de Dados 26
3.3 Tela principal do Supervisório – Laboratório B 28
3.4 processo 29
3.5 Configuração do Clock 30
3.6 Tela Detalhes 31
3.7 Telas de Informações dos dispositivos – TT302 31
3.8 Telas de Informações dos dispositivos – LD302 32
3.9 Janela de Edição de Scripts da Aplicação 33
3.10 Administrador de Fonte de Dados do MS Windows XP 35
3.11 Janela de configuração da nova fonte de dados PostgreSQL ANSI ODBC Driver criada 35
3.12 Config Syscon 37
3.13 Bloco Funcional de Diagnóstico 38
3.14 Bloco Funcional Transdutor 38
3.15 Bloco Funcional AI(Entrada Analógica) 39
3.16 Bloco Funcional de Recursos 39
3.17 Cliente OPC do Elipse SCADA 40
3.18 Tabela BufferEntrada 41
3.19 Tabela Recentes 42
3.20 Tabela Sensores 42

Lista de Tabelas

2.1 Nomenclatura dos principais blocos funcionais do padrão *Foundation Fieldbus* 9

3.1 Tabela de Conversão Inteiro X Unidade de Temperatura. 33
3.2 Tabela de Conversão Inteiro X Unidade de Pressão. 34

Capítulo 1

Introdução

1.1 Sistemas de Automação Industrial

O conceito da automação industrial está relacionado à utilização de várias técnicas destinadas a tornar automáticos diversos processos da indústria. Conseqüentemente, o trabalho braçal do ser humano acaba sendo substituído por equipamentos diversos. O conceito de automação varia com o ambiente e experiência da pessoa envolvida. São exemplos de automação [Maitelli 2003]:

- Para uma dona de casa, a máquina de lavar roupa ou lavar louça.
- Para um empregado da indústria automobilística, pode ser um robô.
- Para uma pessoa comum, pode ser a capacidade de tirar dinheiro do caixa eletrônico.

Esse conceito envolve ainda a idéia de se utilizar potência elétrica ou mecânica para acionar algum tipo de máquina, podendo ser agregada inteligência a tal dispositivo, com o objetivo de proporcionar maior eficiência e segurança na execução de sua tarefa.

A justificativa básica para automatizar um processo industrial, ou uma parte dele, pode se basear nas vantagens apontadas abaixo [Maitelli 2003]:

1. Qualidade: ao se produzir em uma faixa de tolerância à falhas estreita através de um controle de qualidade eficiente, de uma compensação automática de deficiências do processo, ou do uso de tecnologias mais sofisticadas, obtém-se uma qualidade maior na produção;
2. Produtividade: obtida a partir do uso mais eficiente da matéria prima, energia, equipamentos e instalações através, por exemplo, da produção de refugo zero, como conseqüência de uma supervisão da qualidade, na qual o mínimo de matéria prima é desperdiçado;

3. Flexibilidade: consiste na capacidade de admitir com facilidade e rapidez, alterações nos parâmetros do processo de fabricação, em função de inovações freqüentes no produto, do atendimento a especificidades do cliente, ou da produção de pequenos lotes;
4. Viabilidade Técnica: permite que o sistema execute operações impossíveis de realizar por métodos convencionais, como manipulação de componentes extremamente sensíveis e minúsculos.

1.1.1 Breve Retrospecto

O advento da Automação Industrial, por razões etimológicas da expressão, está associado diretamente à necessidade de existir indústria e processos industriais autocontroláveis [Silveira e Lima 2003]. Portanto, pode-se marcar como início da Automação Industrial o século XVIII, com a criação inglesa da máquina a vapor, aumentando a produção de artigos manufaturados, época correspondente à Revolução Industrial. No século seguinte a indústria cresceu e tomou forma, novas fontes de energia e a substituição do ferro pelo aço impulsionaram o desenvolvimento das indústrias na Europa e Estados Unidos. Nesse contexto, nos anos que se seguiram, foram criados dispositivos eletromecânicos chamados relés que, em breve, tomariam as fabricas.

No início do século XX, embora o conceito de indústria já estivesse bastante estabelecido, os ambientes fabris ainda desfrutavam de processos de automação muito rudimentares. Até que Henry Ford teve a grande idéia, a qual mudou o pensamento da indústria contemporânea, propagando-se até os dias de hoje. A idéia revolucionária consistia numa linha de produção, possibilitando uma produção em massa, tornando-se o real gatilho para o grande desenvolvimento industrial do século.

Outro fato marcante na história da automação ocorreu cerca de 40 anos depois com o advento dos Controladores Lógicos Programáveis, cuja criação foi incentivada pela *General Motors*, empresa norte-americana que enfrentava dificuldades com a programação de sua linha de produção. Até então, a programação era toda feita através da utilização de relés e a complexidade de alguns processos produtivos exigia, não raro, instalações em painéis com centenas desses dispositivos. O surgimento dos CLP's ocasionou uma maior flexibilidade, economia e eficiência em tais sistemas, tendo em vista que proporcionavam grandes facilidades na manutenção e uma maior operacionalidade.

Para Constantino Seixas [Filho 2000], o conceito embutido na palavra automação está diretamente ligado a revolução do processo de produção. Primeiro devido a uma especialização causada pela melhor compreensão dos diversos tipos de processo. A automação

de processos contínuos, em batelada e de manufatura requer normas e produtos diferentes, que melhor atendam a identidade de cada setor. A segunda revolução corresponde ao aumento do escopo das atividades. A automação rompeu as barreiras do chão de fábrica e buscou fronteiras mais amplas, se abrangendo a automação do negócio, ao invés da simples automação dos processos e equipamentos.

Com o tempo, os painéis sinópticos e as mesas de controle foram dando lugar ao PC (*Personal Computer*), o qual passou a reinar como a plataforma preferida de supervisão e operação de processos. Os *softwares* SCADA (*Supervisory Control And Data Acquisition*) surgiram em diversos tamanhos, diversos sistemas operacionais e com diversas funcionalidades englobadas. Na área de instrumentação, fez-se necessário uma adequação dos instrumentos para torná-los mais inteligentes. Logo, o padrão para transmissão de sinais analógicos de 4-20mA cedeu espaço para a transmissão digital de dados. Exemplos de alterações que comprovam integralmente o avanço associado a automação industrial.

1.1.2 Níveis de Hierarquia em Sistemas de Automação

A quantidade de níveis hierárquicos dentro de um sistema depende fundamentalmente do tamanho do processo e das necessidades relacionadas à planta industrial [Lima 2004]. Um sistema de automação industrial genérico pode ser caracterizado em seis níveis hierárquicos (Figura 1.1), sendo que em algumas plantas não há a presença de todos.

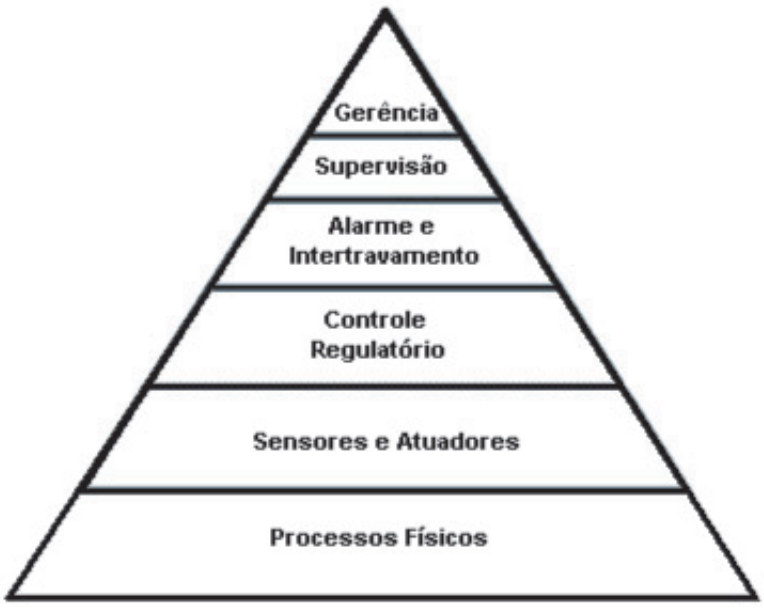


Figura 1.1: Pirâmide hierárquica dos sistemas de automação

Processos Físicos: Consiste na base da pirâmide, estando presente em todos os sistemas de automação. Aqui estão inclusos todos os processos a serem automatizados. Neste nível encontramos o conjunto do conhecimento das técnicas e materiais de produção de uma fábrica: tanques, bombas, caldeiras, robôs, esteiras, motores, entre outros.

Sensores e Atuadores: Camada que contém os dispositivos encarregados de manipular o processo produtivo, tomando as medidas necessárias a partir das informações enviadas pelo nível superior. Por ser a camada mais próxima do processo controlado, também possui a responsabilidade de fazer a aquisição dos dados.

Controle Regulatório: Os sensores e atuadores do nível inferior estão diretamente conectados aos dispositivos desta camada. Os controladores de malha, CLP's, Sistemas Digitais de Controle Distribuído (SDCD), são exemplos de dispositivos que implementam o controle regulatório.

Alarme e Intertravamento: Nível responsável pela segurança do processo como um todo. No momento em que os circuitos de segurança detectam falhas no sistema, os equipamento são levados a um estado seguro (intertravamento), no qual a produção é preservada. Logo em seguida, alarmes são acionados para que os engenheiros de processo e automação possam tomar ações corretivas.

Supervisão: Camada responsável pela emissão de relatórios de operação e exibição de informações. Sua configuração varia desde sistemas mais simples, apenas com interfaces homem-máquina (IHM) locais, até ilhas de supervisão equipadas com computadores poderosos e com os sistemas SCADA.

Gerência: Consiste no nível mais elevado da hierarquia, sendo realizadas análises de dados por um corpo administrativo, o qual articula decisões de caráter econômico e de *marketing*.

1.2 Objetivos e Localização do Trabalho

O objetivo principal do projeto é aproveitar os recursos do Laboratório de Avaliação de Medição em Petróleo (LAMP), para desenvolver um sistema de monitoramento e armazenamento de dados coletados a partir da planta implementada no mesmo.

A planta industrial do LAMP é implementada em escala laboratorial possuindo 6 tanques sendo eles de óleo, água, misturador, auditor, resíduos além de um tanque tratador para separação água/óleo, que possibilita a reutilização da água e do óleo em seguidos testes, sem necessidade de descartes a cada teste. A automação é feita com avançada tecnologia de barramento de campo. A principal característica do laboratório é a integração

de três destas tecnologias: *Foundation Fieldbus* , MODBus RTU e ponto-a-ponto(Serial).

Utilizando tais recursos deverá ser criada uma arquitetura unindo software e hardware visando monitorar não somente as características do processo mas também dos dispositivos, de modo a criar uma base de dados completa sobre diversas variáveis intrínsecas aos dispositivos para estudo do seu comportamento passado, presente e futuro.

1.3 Estrutura do Trabalho

Este documento foi dividido em quatro Capítulos, os quais apresentam desde uma visão geral dos Sistemas de Automação Industrial até a proposta e os resultados obtidos no trabalho.

Dessa forma, no segundo capítulo é construída toda a fundamentação teórica para o desenvolvimento do trabalho. Em primeiro lugar, faz-se uma análise da comunicação entre processos industriais enfatizando a explicação do protocolo *Foundation Fieldbus* de comunicação, em seguida explica-se o protocolo OPC, sistemas supervisórios, e por fim, são apresentados os conceitos básicos do Sistema Gerenciador de Base de Dados Relacional.

O terceiro capítulo apresenta toda a proposta de trabalho, com sua implementação e resultados.

No último capítulo são discutidos os resultados alcançados com a implementação e uma opinião sobre o funcionamento do sistema como um todo.

Capítulo 2

Fundamentação Teórica

Neste capítulo serão descritos os conhecimentos teóricos levados em conta para o desenvolvimento deste trabalho, sendo eles: Comunicação entre Processos Industriais – Protocolo *Foundation Fieldbus* , Padrão OPC, Sistemas Supervisórios – Elipse Software e Sistemas de Gerenciamento de Banco de Dados – PostgreSQL .

2.1 Comunicação entre Processos Industriais – Protocolo *Foundation Fieldbus*

A rede Foundation Fieldbus (FF) é um sistema de comunicação digital, serial e bi-direcional, que funciona como uma rede local para instrumentos usados em processos e automação industrial, com capacidade embutida para distribuir o controle de aplicação através da rede industrial. Ela também pode ser interligada a redes TCP/IP/Ethernet, com o intuito de configuração remota de dispositivos. A estratégia de controle distribuído ao longo dos dispositivos de campo é possível porque todos os dispositivos possuem micro-processadores e memória com várias funções, inclusive a estratégia de controle PID, e alguns fabricantes já disponibilizam controle fuzzy e outros tipos de estratégias de controle. Graças a todas essas novas possibilidades, o conceito de gerenciamento de processos atualmente permite novas tarefas de automação, como: novas configurações, diagnósticos de desempenho em tempo real e manutenção de registros e ferramentas.

2.1.1 Características da Tecnologia

A *Foundation Fieldbus* possui um protocolo confiável e determinístico para comunicação em instrumentação e controle de processos, interligando equipamentos, como: sensores, atuadores e controladores, com a habilidade de operar dispositivos múltiplos,

independentemente do fabricante, no mesmo sistema sem a mínima perda de funcionalidade e interoperabilidade, desde que não sejam utilizados funcionalidades específicas de um fabricante.[Lima 2004]

O modelo de referência de comunicação em camadas (modelo OSI) é utilizado para modelar os componentes fundamentais da tecnologia *Foundation Fieldbus* (Figura 2.1) nos três seguintes componentes:

- Camada física;
- Camada de comunicação, e;
- Camada de aplicação ao usuário.

Os níveis 3 ao 6 do modelo OSI não são implementados na tecnologia *Foundation Fieldbus* pois se trata de níveis utilizados na rede local.

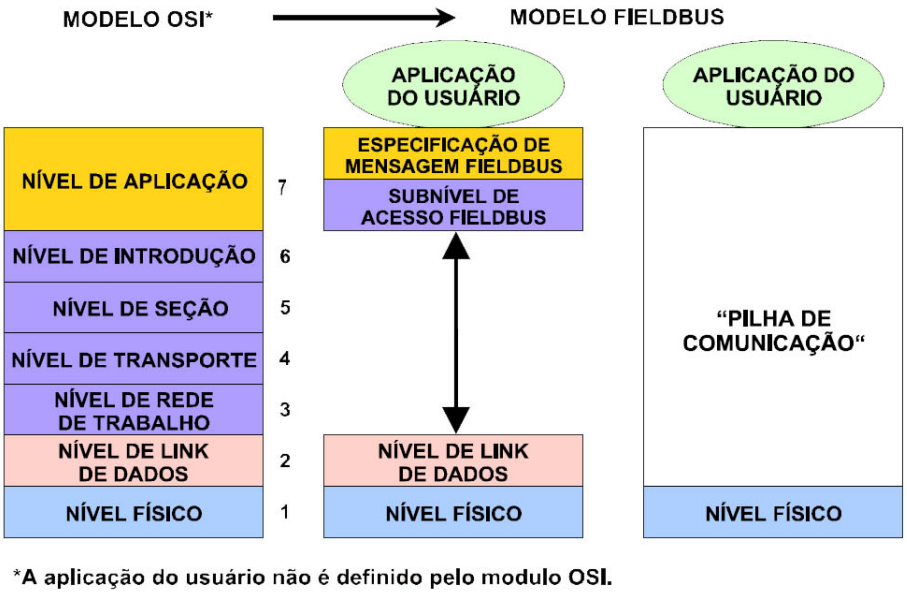


Figura 2.1: Modelo OSI X *Foundation Fieldbus*

2.1.2 Camada Física

A camada física equivale ao nível físico do modelo OSI. No nível físico, os sinais *Foundation Fieldbus* , padronizados pelo IEC (*International Engineering Consortium*) e pela ISA (*The Instrumentation, Systems and Automation Society*), são codificados usando a codificação *Manchester Biphase-L*. Este tipo de sinal carrega junto com os dados a informação de clock para sincronização.

Os dados da tecnologia *Foundation Fieldbus* podem trafegar junto da energia que alimenta os dispositivos, necessitando, o instrumento, então apenas de um par de fios, que pode ser o mesmo usado em dispositivos 4-20mA. O dispositivo transmissor entrega +10mA a 31.25Kbps para uma carga de até 50Ω para criar uma tensão de 1V pico-a-pico modulada acima da corrente direta da fonte de tensão. Para algumas aplicações, a tensão pode variar de 9 a 32V. O comprimento do cabo é determinado pela taxa de comunicação, tipo e tamanho deste e potência da linha.[Lima 2004]

2.1.3 Camada de Comunicação

A camada de comunicação possui basicamente três subcamadas: a subcamada inferior de enlace de dados (controle de erro e política de acesso ao meio), que faz interface com a camada física; a subcamada intermediária de acesso a serviços *fieldbus*(*FAS – Fieldbus Access Sublayer*) e a subcamada superior de montagem de mensagens (*FMS -Fieldbus Message Specification*).

A tecnologia *Foundation Fieldbus* define dois tipos básicos de equipamentos disponíveis na camada de comunicação:

- Dispositivos básicos, que são os sensores, atuadores, entre outros;
- Dispositivos de Link Mestre, que preferencialmente será um LAS (*Link Active Scheduler*).

Um LAS é um dispositivo que controla de forma determinística os tempos que os dispositivos transmitem (publicação) os dados dos *buffers* para a rede. Quem estiver configurado para receber (assinante) copia estes dados. Geralmente o LAS é implementado num dispositivo especial denominado de *Linking Device*. Porém, na sua ausência, qualquer outro dispositivo pode desempenhar o papel do LAS, de modo a não parar o funcionamento da rede. Entre as transmissões de mensagens agendadas também podem transitar mensagens de forma não agendada.

O LAS concede permissão para um dispositivo usar o barramento emitindo um sinal de passagem de token (PT). Ao receber este sinal, o dispositivo transmite, se necessitar. Isso significa que esta tecnologia funciona como o protocolo passagem do *token* de barramento.[Lima 2004]

2.1.4 Camada do Usuário

Uma característica única da *Foundation Fieldbus* , que assegura interoperabilidade de dispositivos, é o uso de uma Camada de Usuário, padronizada e completamente especifi-

cada, baseada em blocos e tecnologia de descrição de dispositivos. A Camada de Usuário define um processo de aplicação de blocos de função usando blocos de recursos, blocos de função, blocos transdutores, gerenciamento de sistema e de rede e tecnologia de descrição de dispositivos.

Blocos de Recursos: Definem parâmetros que são necessários a qualquer aplicação.(Exemplo: Número serial de fabricação.)

Blocos de Função: Encapsulam funções de controle. (Exemplos: Controlador PID, Entrada Analógica, etc.)

Blocos Transdutores: Representam uma interface para sensores, tais como: de temperatura, pressão e fluxo.

Blocos Transdutores	
Analog Input (AI)	Bloco de entrada de dados analógico.
Analog Output (AO)	Bloco de saída de dados analógico.
Transducer	Bloco conversor de grandezas físicas.
Bloco de Função	
PID	Bloco controlador de ação proporcional, integrativa e derivativa.
Blocos de Recursos	
Resource	Bloco de recursos dos instrumentos.
Display	Bloco de apresentação de informações no display.

Tabela 2.1: Nomenclatura dos principais blocos funcionais do padrão *Foundation Fieldbus* .

A Tabela 2.1 apresenta a nomenclatura de alguns dos principais blocos funcionais padronizados pela FF. Os blocos de função são incorporados dentro de equipamentos *Foundation Fieldbus* para conseguir a funcionalidade desejada do dispositivo, bem como definir uma vasta faixa de características e comportamentos que devem trabalhar de maneira padrão para que os dispositivos possam interoperarem.

2.2 *OLE For Process Control - OPC*

Em 1995, algumas empresas, *Fisher-Rosemount*, *Rockwell Software*, *Opto 22*, *Intellution*, e *Intuitive Technology*, se reuniram com o objetivo de desenvolver um padrão baseado na tecnologia OLE/DCOM para acesso à dados de tempo real dentro do sistema operacional Windows. Neste trabalho foram envolvidos membros da *Microsoft* para suporte técnico à solução a ser adotada. Este grupo sem fins lucrativos é formado por diversas empresas e é gerenciado pela organização OPC Foundation. Basicamente, o padrão

OPC estabelece as regras para que sejam desenvolvidos sistemas com interfaces padrões para comunicação dos dispositivos de campo (CLPs, sensores, balanças, etc.) com sistemas de monitoração, supervisão e gerenciamento (SCADA – *Supervisory Control and Data Aquisition*, Sistemas de Controle e Aquisição de Dados, MES – *Manufacturing Execution Systems*, Sistemas de Execução da Manufatura, ERP – *Enterprise Resource Planning*, SIGE - Sistemas Integrados de Gestão Empresarial, etc.).

2.2.1 Histórico

A primeira especificação produzida pelo grupo foi publicada em agosto de 1996, chamada OPC *Specification Version* 1.0. O principal objetivo do grupo é atender às necessidades da indústria, através do aprimoramento e ampliação da especificação OPC. A estratégia adotada foi a criação de extensões à especificação existente, definição da adição de novas especificações e a realização de modificações para manter a compatibilidade máxima com as versões existentes. Em setembro de 1997 foi liberada a primeira atualização da especificação OPC que passou a ser chamada de OPC *Data Access Specification Version* 1.0A. Atualmente existem as seguintes especificações publicadas ou em processo de aprovação:

- ***OPC Overview*** (Versão 1.00) - Descrição geral dos campos de aplicação das especificações OPC;
- ***OPC Common Definitions and Interfaces*** (Versão 1.00) - Definição das funcionalidades básicas para as demais especificações;
- ***OPC Data Access Specification*** (Versão 2.05) - Definição da interface para leitura e escrita de dados de tempo real;
- ***OPC Alarms and Events Specification*** (Versão 1.02) - Definição da interface para monitoração de eventos;
- ***OPC Historical Data Access Specification*** (Versão 1.01) - Definição da interface para acesso a dados históricos;
- ***OPC Batch Specification*** (Versão 2.00) - Definição da interface para acesso aos dados de processos por batelada (batch).Esta especificação é uma extensão da *OPC Data Access Specification*;
- ***OPC Security Specification*** (Versão 1.00) - Definição da interface para utilização de políticas de segurança;
- ***OPC and XML*** (Versão candidata 1.05) - Integração entre OPC e XML para aplicações via Internet (web). Está em fase de elaboração a especificação *OPC DX Data*

Exchange for Ethernet. Esta especificação tem a finalidade de definir a comunicação entre diferentes servidores conectados através de uma rede *Ethernet* TCP/IP.

Até então, esta comunicação não é possível sem a utilização de um cliente e uma aplicação para intermediar a troca de dados, o servidor.

2.2.2 Interfaces de Acesso ao Servidor

Existem duas possíveis interfaces de acesso aos servidores OPC, interfaces do tipo *custom* e interfaces do tipo *automation*. A interface *custom* define o acesso aos servidores OPC por aplicações clientes desenvolvidas através de linguagens que suportam as chamadas das funções por ponteiros, tais como C/C++, Delphi, etc. Entretanto, existem linguagens tais como VisualBasic e VBA que não suportam ponteiros para funções. Neste caso foi introduzido o conceito da interface tipo *automation*. Através da interface tipo automation, os clientes desenvolvidos nestas linguagens podem fazer uso de uma interface padrão onde os métodos são chamados pelo nome e não por ponteiros. Existem portanto, especificações OPC separadas para interfaces do tipo *custom* e *automation*.

A publicação das especificações para o padrão OPC possibilitou o desenvolvimento de diversos produtos para automação industrial, os quais se beneficiam das vantagens proporcionadas pelo padrão:

- Padronização das interfaces de comunicação entre os servidores e clientes de dados de tempo real, facilitando a integração e manutenção dos sistemas;
- Eliminação da necessidade de *drivers* de comunicação Específicos (proprietários);
- Melhoria do desempenho e otimização da comunicação entre dispositivos de automação;
- Interoperabilidade entre sistemas de diversos fabricantes;
- Integração com sistemas MES, ERP e aplicações Windows (Excel, etc.);
- Facilidade de desenvolvimento e manutenção de sistemas e produtos para comunicação em tempo real;
- Facilidade de treinamento.

Atualmente existem diversos produtos no mercado que utilizam o OPC para comunicação com dispositivos de chão de fábrica. O OPC está se tornando rapidamente o padrão de comunicação adotado pelo mercado de automação industrial e pela indústria.

2.2.3 Aspectos Práticos para Utilização do Padrão OPC

Para a especificação e utilização do padrão OPC, necessita-se estar ciente de alguns pontos chaves para o perfeito entendimento de como se beneficiar do uso da comunicação OPC. Por isso, o estudo das especificações torna-se um processo difícil, uma vez que as mesmas são direcionadas para desenvolvedores e programadores, sendo necessário o conhecimento prévio de linguagens e ambientes de desenvolvimento. Para simplificar o entendimento do padrão OPC, estes pontos são apresentados a seguir.

Plataforma Windows ou não?

Basicamente, o padrão OPC é nativo da plataforma Windows. Dentro desta plataforma, existem variações para as versões do Windows (CE, 9X, NT, 2000 e XP), mas para todas estas é possível a comunicação OPC. Para plataformas não-Windows, existem algumas soluções que consistem em portar o DCOM para estas plataformas. No futuro, a especificação OPC para XML deverá facilitar a integração de plataformas não-Windows para a comunicação OPC.

Cliente ou Servidor OPC?

As aplicações e produtos existentes no mercado podem ser somente um cliente, um servidor ou ambos, isto varia de caso a caso. Normalmente, os produtos para monitoração de dados (IHM's; sistemas supervisórios, etc.) são clientes OPC. Já os produtos que fazem a comunicação direta com os dispositivos de campo utilizando protocolos proprietários são servidores OPC. Cada produto pode incorporar as duas funcionalidades, sendo o mais comum que uma aplicação normalmente cliente possa ser servidor, e não o contrário.

Número de Clientes x Número de Servidores

O número de servidores OPC necessários para uma determinada aplicação irá depender do produto a ser utilizado. Normalmente, os fabricantes de dispositivos de campo (CLPs; dispositivos inteligentes, etc.) fornecem um servidor OPC capaz de comunicar com todos os protocolos dos sua linha produtos . Este servidor é um software para o ambiente Windows que é executado em um microcomputador, normalmente PC. Ou seja, um servidor OPC da Rockwell, o RSLinx por exemplo, permite que diversos drivers de comunicação sejam configurados para as diversas redes (ControlNet, DeviceNet, Ethernet, DH+, etc.), algumas para as aplicações clientes, outras para as fontes de dados. Neste

caso, o RSLinx funciona como um único servidor OPC, capaz de comunicar com diversos clientes OPC sendo executados na mesma máquina ou em máquinas remotas. Existem servidores OPC de terceiros que permitem que sejam configurados drivers de comunicação para diversas redes e protocolos de diferentes fabricantes. Como exemplo podemos citar os servidores da Kepware e da Matrikon. Neste caso, um único produto poderá fornecer dados de diferentes fabricantes. Cada cliente OPC pode conectar-se à diferentes servidores, os quais podem estar processando na mesma máquina ou remotamente em máquinas diferentes. Portanto, qualquer produto que funcione como cliente OPC poderá se comunicar com quaisquer servidores OPC de quaisquer fabricantes.

Formato de Dados OPC (*Time Stamp* e Qualidade)

Pela especificação do padrão, todo servidor de dados deve enviar o dado OPC no formato apresentado a seguir:

- **Valor do dado:** Todos os tipos de dados VARIANT definidos pela interface DCOM são suportados;
- ***Time Stamp*:** Esta informação é fornecida pelo servidor através da leitura do *time stamp* dos dispositivos de campo ou por geração interna. É utilizada a estrutura padrão do Windows para o UTC (*Universal Time Coordinated*).
- **Informação de estado:** São reservados 2 *bytes* para codificação do estado do dado fornecido pelo servidor. Por enquanto, apenas o uso do *byte* menos significativo foi definido. Dois *bits* definem a qualidade do dado que pode ser:
  - *Good*: Dado válido;
  - *Bad*: No caso de perda do *link* de comunicação com o dispositivo de campo, sem comunicação, por exemplo;
  - *Uncertain*: No caso de existir o *link* de comunicação mas o dispositivo de campo estiver fora de operação.

Quatro bits fornecem um detalhamento do estado apresentado, tais como *NotConnected* e *Last Usable Value*. Os últimos dois bits podem conter dados de diagnóstico no caso de falha de um sensor, por exemplo.

Configuração dos dados OPC no Cliente

Normalmente, os produtos de mercado não permitem muita flexibilidade para a configuração dos dados solicitados pelo cliente. Isto se deve basicamente à preservação da cultura anterior para os drivers de comunicação específicos. Considerando o caso mais

comum que consiste nos servidores de dados OPC(*OPC Data Access*), os clientes podem definir basicamente as seguintes configurações:

- **Criação de grupos e itens OPC**: Basicamente, todos os dados OPC são chamados de itens. Cada item pode ser de um tipo diferente de dado compatível com a especificação OPC. Os diversos itens são organizados em grupos OPC, os quais definem as principais características de leitura dos itens (Taxa de Atualização, Estado Ativo/Inativo, Banda Morta, Leitura Síncrona/Assíncrona).
- **Leitura Síncrona ou Assíncrona**: Para um determinado grupo OPC pode ser definido se a leitura dos dados é feita de forma síncrona, a qual depende de uma confirmação de execução antes de uma nova leitura, ou assíncrona, a qual não depende da confirmação. Normalmente é utilizada a leitura assíncrona, a qual garante um melhor desempenho.
- **Leitura de dados direto do dispositivo**: A partir da versão 2.0 da especificação para o servidor de dados, é possível fazer a seleção no cliente OPC para leitura dos dados da memória *cache* do servidor ou diretamente do dispositivo de campo.
- **Estado Ativo/Inativo**: Cada item ou grupo pode ter o seu estado alterado pelo cliente para Ativo, habilitando a comunicação do mesmo, ou Inativo.
- **Leitura Cíclica ou por Mudança de Estado**: O cliente OPC pode definir se os dados do servidor serão lidos de forma cíclica ou por mudança (transição) de estado. Na leitura cíclica, o cliente faz a requisição de leitura regularmente, independentemente se os dados sofreram alteração de valor ou não. No caso de leitura por mudança de estado, o servidor fica responsável por enviar para os clientes os itens que sofrerem alteração de seu estado (Qualidade do dado) ou quando os valores dos itens de um determinado grupo ultrapassarem o valor da banda morta.
- **Banda Morta**: É utilizada para definir os valores limites de transição para os itens de um determinado grupo, para os quais o servidor fará o envio para os clientes quando a alteração dos valores dos itens estiver fora da banda especificada.
- **Escrita de dados OPC**: A escrita de dados OPC funciona de forma independente da leitura. Assim como na leitura, a escrita pode ser síncrona ou assíncrona. Entretanto, os comandos de escrita são executados imediatamente pelo servidor, sendo enviados diretamente para os dispositivos de campo. Está prevista para a versão 3.0 do servidor de dados, a possibilidade de se fazer a escrita de dados na memória *cache* do servidor e depois a transferência cíclica dos dados para os dispositivos de campo. Este recurso será muito útil para os dispositivos que dependem de comandos igualmente espaçados no tempo, tal como os sistemas de controle de

movimento.

- **Comunicação de Blocos de Dados**: O padrão OPC permite a comunicação de blocos de dados (vetores) entre o servidor e os clientes. Isto representa uma grande otimização, pois as informações de *time stamp* e estado do dado são tratados e fornecidos apenas uma vez para um conjunto de dados, reduzindo assim o *overhead* da comunicação. Neste caso, cada item é configurado como um bloco de dados.
- **Redundância com OPC**: As especificações do padrão OPC não fazem menção à utilização de servidores redundantes. Entretanto, cada cliente OPC pode implementar facilmente um mecanismo para conexão simultânea em mais de um servidor, verificação do estado do servidor e ativação/desativação dos grupos para o servidor que estiver funcionando. Esta solução é encontrada apenas em alguns produtos, não sendo regra geral a disponibilização deste recurso para a maioria dos produtos de mercado. O produto ORB (*OPC Redundancy Broker*) da Matrikon permite que clientes comuns possam fazer o chaveamento para servidores redundantes.
- **Desempenho da comunicação OPC**: Em linhas gerais, o desempenho da comunicação OPC se aproxima do desempenho apresentado por sistemas que utilizam *drivers* de comunicação específicos e otimizados. Normalmente, os *drivers* específicos possuem um ótimo desempenho após serem devidamente depurados e otimizados, o que via de regra não acontece em muitos casos. Como um servidor OPC nada mais é do que uma camada de software a mais para implementar as interfaces padrões e os mecanismos de comunicação com o cliente, é de se esperar que o desempenho do mesmo só seja afetado em relação à comunicação com o cliente e não com o dispositivo de campo. No caso da comunicação com o dispositivo de campo, cada fornecedor pode implementar o *driver* e o protocolo que melhor se ajustem às necessidades do dispositivo e da rede de comunicação. Desta forma, o desempenho do servidor OPC está mais relacionado à capacidade dos recursos de hardware da máquina que executa a aplicação do servidor do que propriamente do *driver* específico. Como os recursos de hardware estão cada vez mais poderosos em relação à capacidade de processamento, isto não tem se mostrado como um problema real. Entretanto, o que se tem verificado na prática é que muitos clientes e servidores OPC não implementam a comunicação de blocos de dados, fazendo a leitura de itens separadamente, o que ocasiona um grande *overhead* devido ao tratamento separado de *time stamp* e estado do dado para cada item OPC. Outro ponto importante que muitos clientes OPC não implementam, consiste no agrupamento de dados que precisam ser lidos sob demanda, tais como animações de telas sinópticas, janelas de operação de equipamentos, relatórios, etc. Os dados necessários

para estes elementos (objetos) de monitoração, normalmente podem ser lidos sob demanda, de forma que somente quando o objeto estiver selecionado, será ativado o grupo OPC no servidor para leitura dos dados. Quando o objeto não estiver selecionado, o grupo OPC ficará desativado, fazendo com os dados não sejam lidos e melhorando o desempenho da comunicação.

- **Segurança para acesso ao sistema**: Para a implementação do controle de acesso ao servidor OPC podem ser utilizados dois métodos. O método normalmente usado consiste nos mecanismos proporcionados pelo próprio DCOM, os quais são configurados no Windows NT executando-se o comando DCOMCNFG. Outra forma menos usual consiste em se utilizar mecanismos implementados pelo cliente e servidor conforme a especificação do padrão OPC. O controle de acesso é fundamental para o caso de acesso remoto e para a comunicação via Internet prevista com a especificação do OPC com XML.[LAMP 2003]

2.3 Sistemas Supervisórios

Os sistemas supervisórios são *softwares* que permitem o monitoramento de informações e variáveis de um processo produtivo ou instalações físicas, por exemplo. Essas informações podem ser obtidas através de equipamentos de aquisição de dados e, posteriormente, podem ser manipuladas, analisadas, armazenadas e, em seguida, apresentadas ao usuário através de uma interface amigável, condicionando a supervisão e o controle de uma planta automatizada. Os sistemas supervisórios são também conhecidos como sistemas SCADA(Sistemas de Controle e Aquisição de Dados), ou ainda, Interface Homem-Máquina.

Inicialmente, os sistemas SCADA permitiam o monitoramento de sinais representativos de medidas e estados de dispositivos. Tais informações eram apresentadas em um painel de lâmpadas e indicadores, sendo permitido o controle desses estados pelo operador apenas com uso de botoeiras.

Hoje em dia, os sistemas de automação industrial são beneficiados por várias tecnologias de computação e comunicação, permitindo que o monitoramento e controle dos processos industriais complexos sejam realizados de locais distantes, e sua apresentação mostrada de forma amigável para os operadores com recursos gráficos e multimídia.

Para obtenção desses resultados, o sistema SCADA faz a identificação das variáveis da aplicação através de *tags*, as quais podem exercer várias funções computacionais, ou ainda, fazer a representação de entradas/saídas de dados do processo industrial controlado, correspondendo às variáveis do processo real, como temperatura, nível e vazão. Dessa

forma, cria-se o elo entre o controlador e o sistema. A apresentação dos dados adquiridos só é possível a partir da leitura dos valores associados às *tags*. Os sistemas SCADA podem verificar o estado das variáveis e fazer com que sejam obtidas condições de alarmes, nas quais o valor da *tag* ultrapassa um limite ou condição pré-estabelecida, podendo ainda, fazer a gravação destes registros em Bancos de Dados, ativação de som, mensagem ou mudança de cores dos indicadores nas telas.

2.3.1 Componentes Físicos de um Sistema de Supervisão

Os Componentes físicos de um sistema de supervisão, de forma simplificada, são: sensores e atuadores, redes de comunicação, estações remotas (aquisição/controle) e de monitoração central (sistema SCADA) [Silva e Salvador 2005].

Os sensores são elementos que sentem a variável a ser medida. Os transmissores condicionam o sinal do sensor e o convertem para um sinal adequado para a transmissão aos controladores. Esses sinais de transmissão são normalmente elétricos e podem ser classificados em digitais e analógicos. Já os atuadores são responsáveis por executar as tarefas enviadas pelo controlador e permitem o controle da variável de processo [Souza 2005].

O controle e aquisição de dados estão presentes nas estações remotas, CLP's(Controladores Lógico Programáveis) e RTU's (*Remote Terminal Units*), nas quais os dados são lidos através da instrumentação do nível inferior e procedimentos são executados através dos atuadores [Lima 2004]. Os CLP's e RTU's são equipamentos utilizados, por exemplo, nas fábricas com o objetivo de obter entradas, realizar cálculos ou controles, e atualizar suas saídas.

Na rede de comunicação, temos a camada por onde as informações fluem dos CLP's/RTU's para o sistema SCADA.

Nas estações de monitoramento central encontram-se as principais partes dos sistemas SCADA, as quais são responsáveis por todo o monitoramento e apresentação dos dados gerados nas estações remotas e realizar as ações conforme os alarmes detectados, e ainda, podem permitir o compartilhamento de informações coletadas, optando por uma arquitetura distribuída em uma rede de computadores ou centralizada em um único computador [Silva e Salvador 2005].

Um exemplo dos componentes físicos de um Sistema de Supervisão podem ser observados na figura 2.2.

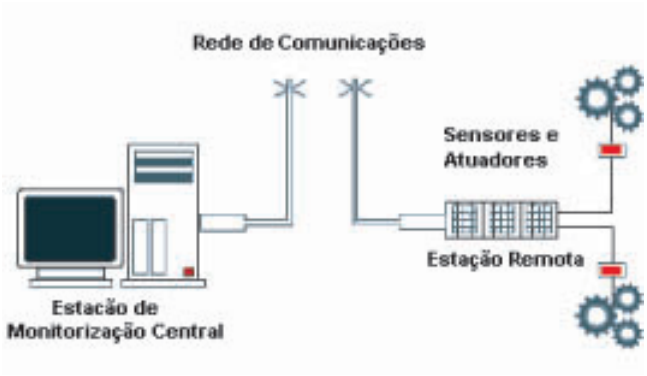


Figura 2.2: Sistema de supervisão e controle

2.3.2 Componentes Lógicos de um Sistema SCADA

Os sistemas SCADA, normalmente, têm suas principais funções organizadas em blocos ou módulos que irão permitir uma alta ou baixa flexibilidade e robustez, dependendo do tipo de atividade que venha a ser utilizada.

Resumindo, podemos descrever essas funções como [Silva e Salvador 2005]:

- Núcleo de processamento;
- Comunicação com CLP's/RTU's;
- Gerenciamento de Alarmes;
- Históricos e Banco de Dados;
- Lógicas de programação interna (*Scripts*) ou controle;
- Interface gráfica;
- Relatórios;
- Comunicação com outras estações SCADA;
- Comunicação com Sistemas Externos/Corporativos;
- Outros.

O funcionamento de um sistema SCADA é feito basicamente através da comunicação dos equipamentos de campo com o núcleo de processamento principal do *software* SCADA. A partir daí, este núcleo principal fica responsável pela disseminação e coordenação das informações para os demais sistemas integrados. Essas informações chegam ao sistema e são mostrados na forma de gráficos, animações, relatórios, facilitando ao operador do sistema o acompanhamento das operações. Além disso, podem ser exibidos a evolução do estado de certos dispositivos e do processo supervisionado, podendo mostrar alarmes e indicar ações a serem tomadas, ou ainda, interferir no processo por medida de segurança.

Novas tecnologias computacionais estão surgindo e permitindo o desenvolvimento de sistemas SCADA cada vez mais confiáveis e flexíveis, incluindo a diminuição do tempo gasto na configuração dos sistemas às necessidades do processo.

2.3.3 Modos de Comunicação

A função principal de um sistema SCADA consiste na troca de informações, podendo ser sintetizada das seguintes formas [Silva e Salvador 2005]:

- Comunicação com os CLP's/RTU's;
- Comunicação com outras estações SCADA;
- Comunicação com outros sistemas.

Os dispositivos de campo podem se comunicar com o supervisório através de um protocolos comum, cuja metodologia pode ser tanto de caráter aberto quanto de caráter fechado.

A comunicação entre sistemas SCADA pode ocorrer entre protocolos criados pelos próprios fabricantes destes sistemas ou por protocolos já desenvolvidos como, por exemplo, rede *Ethernet TCP/IP*, ou linhas discadas.

O meio de comunicação cada vez mais usado para sistemas SCADA é a Internet. A Internet é utilizada através de tecnologias e padrões como *Ethernet*, TCP/IP, HTTP, HTML, sendo possível a comunicação e compartilhamento de informações entre a área de produção e a área de supervisão e controle das estações (sistemas). Essa comunicação é provida através de um *browser* de Internet, no qual pode-se controlar em tempo real, uma máquina localizada em qualquer parte do mundo. O *browser* faz uma solicitação de uma operação, através do protocolo HTTP, ao servidor *web*, e logo depois recebe uma resposta em forma de uma página HTML. A grande vantagem do *browser* como interface de visualização SCADA é o modo simples de interação, ao qual a maioria das pessoas já está habituada [Silva e Salvador 2005].

2.3.4 Sistema Supervisório Elipse SCADA

O Elipse Scada é um software utilizado no desenvolvimento de sistemas de supervisão e controle de processos. Através da coleta de informações de qualquer tipo de equipamento, os operadores podem monitorar e controlar com precisão todos os processos do chão de fábrica, bem como máquinas e recursos, gerenciando de forma rápida e eficiente toda a produção. Dados em tempo real são apresentados de forma gráfica, permitindo o

tratamento das informações de várias maneiras, como o armazenamento histórico, a geração de relatórios e a conexão remota, entre outras possibilidades. Análises precisas e um rápido tempo de resposta resultam em menos perdas e altos níveis de qualidade.

Utilizando o Organizer (Árvore do Aplicativo) pode-se, no modo off-line, criar, organizar e documentar uma aplicação de maneira muito simples e fácil. Com ele é possível acessar todos os elementos de sistema e suas propriedades, tendo-se assim uma visão geral do aplicativo. Através da ferramenta de configuração on-line é possível, por exemplo, alterar a cor, tamanho e posição de uma janela.

O usuário ainda pode escolher entre utilizar o mouse, teclado ou touchscreen para operar o sistema de supervisão.

Versões

O Elipse Scada é constituído por quatro versões distintas indicadas segundo as necessidades do usuário: Elipse Scada VIEW, Elipse Scada MMI, Elipse Scada PRO (Professional) e Elipse Scada CE. Abaixo são descritas as principais características de cada uma.

A primeira delas é a versão View, indicada para aplicações simples, de interface com o operador para monitoração e acionamentos. As informações recebidas pelo View estão disponíveis também para outras aplicações que possam trabalhar com NetDDE (troca dinâmica de dados em rede) trabalhando como servidor DDE(*Dynamic Data Exchange*). Nesta versão estão disponíveis funções que permite o monitoramento e controle, comunicação com CLP via drivers DLL(*Dynamic-link library*), objetos de tela para a produção de interfaces, como por exemplo, botão, medidores (gauges), caixa de texto, gráfico de barra e tendência, imagem, animações e alarmes, importação de imagens de editores gráficos, controle de acesso através de lista de usuários (autenticação), comunicação em bloco, uso de scripts (ferramenta fundamental em um software supervisório), servidores e cliente DDE, número ilimitado de tags e servidor de aplicações remotas.

A segunda delas é a versão MMI (Man Machine Interface) é indicada para aplicações de médio porte, onde é necessário o armazenamento de dados, tratamento de informações e criação de relatórios complexos. Além das funcionalidades incorporadas na versão View, esta versão conta ainda com: históricos, receitas, relatórios, suporte a CEP (controle estatístico de processos), alarmes tipo histórico e browser (que são objetos de tela) e função de gerar log de alarmes em disco.

Já a versão Professional é indicada para aplicações de qualquer porte, que envolvam comunicação em rede, local ou remota, ou ainda que necessite a troca de informações

entre banco de dados via ODBC(Open Database Connectivity). Além das funcionalidades incorporadas na versão MMI, esta versão inclui ainda ODBC, suporte a DAO (Data acess objects), cliente e servidor de rede. Neste trabalho essa será a versão utilizada, por termos a disposição um *hardkey*.

Por fim, a versão CE permite executar aplicações Elipse SCADA em dispositivos baseados no sistema operacional Windows CE, como IHMs, dispositivos sem disco em geral e outros dispositivos móveis. O Elipse SCADA CE não comporta todas as funcionalidades dos pacotes anteriores.

Módulos de Operação

O Elipse Scada possui três módulos para sua operação: Configurador, Runtime e Master. O módulo ativo é definido a partir de um dispositivo de proteção(*hardkey*) que é acoplado ao computador. Enquanto que os módulos Configurador e Master foram especialmente desenvolvidos para a criação e o desenvolvimento de aplicativos, o módulo Runtime permite apenas a execução destes. Neste módulo, não é possível qualquer alteração no aplicativo por parte do usuário.

Na ausência do *hardkey*, o software pode ainda ser executado em modo Demostração. Como não necessita de hardkey, o modo Demo pode ser utilizado para a avaliação de software. Ele possui todos os recursos existentes no módulo Configurador, com exceção de que trabalha com um máximo de 20 *tags* (variáveis de processo) e permite a comunicação com equipamentos de aquisição de dados por até 2 horas. Neste modo, o software pode ser livremente reproduzido e distribuído.

Os módulos Runtime e Master estão também disponíveis em versões Lite que possuem as mesmas características, porém são limitadas em número de tags: Lite 75 com 75 tags e Lite 300 com 300 tags.

Recursos

Os principais recursos do Elipse Scada, são: Geração de Históricos, Relatórios, Supervisão e controle de estações à distância, Cross-Reference, acesso a Banco de Dados, Drivers de comunicação e OPC, Conexão remota com CLPs, Alarmes, Lógicas (Scripts) e Ferramentas de depuração.[Elipse 2007]

2.4 Sistema Gerenciador de Base de Dados Relacional – PostgreSQL

Um SGBDR (Sistema Gerenciador de Base de Dados Relacional), permite que bancos de dados sejam concorrentemente partilhados por inúmeros usuários e aplicações. É sustentado pelo modelo relacional, que é completo e consistente[Neves 2002].

O SGBDR PostgreSQL é Open Source e implementa os padrões SQL ANSI 92, 96 e 99. Permite a utilização de gatilhos, visões, procedimentos armazendos, e muitas outras funcionalidades dos bancos de dados mais conhecidos do mercado, além de possuir *drivers* ODBC e JDBC para interface com linguagens de programação.

Stored Procedures (Procedimentos Armazenados)

Procedimento armazenado ou Stored Procedure é uma coleção de comandos em SQL para gerenciamento de Banco de dados. Encapsula tarefas repetitivas, aceita parâmetros de entrada e retorna um valor de status (para indicar aceitação ou falha na execução). O procedimento armazenado pode reduzir o tráfego na rede, melhorar a performance, criar mecanismos de segurança, etc.

Triggers (Gatilhos)

Gatilho ou trigger é um recurso de programação presente na maioria dos sistemas de gerenciamento de banco de dados, utilizado para associar um procedimento armazenado a um evento do banco de dados (inclusão, exclusão, atualização de registro, por exemplo) de modo que o procedimento armazenado seja executado automaticamente sempre que o evento associado ocorrer. É muito utilizada para ajudar a manter a consistência dos dados ou para propagar alterações em um determinado dado de uma tabela para outras[Neves 2002].

SQL

O SQL (*Structured Query Language*) é a linguagem estruturada para acessar dados num SGBD relacional. Essa linguagem foi criada pela IBM no início dos anos 80 e é utilizada até hoje, com algumas modificações ao longo do tempo.

As declarações da linguagem SQL estão divididas em duas categorias funcionais: DDL (*Data Definition Language*) e DML (*Data Manipulation Language*). A DDL é responsável pelas instruções de criação e manipulação de entidades estruturais, como,

por exemplo, as tabelas, enquanto a DML é responsável pelas instruções de manipulação de dados contidas no SGBDR[Neto 2003].

Da DDL fazem parte declarações como, por exemplo:

- **ALTER TABLE:** Modifica a estrutura de uma tabela. Esse comando tem como principais funções adicionar novas colunas ou renomear colunas já existentes e também o seu próprio nome;
- **CREATE TABLE:** Esse comando cria uma nova tabela no banco de dados corrente. Uma observação a ser feita é que não podemos criar uma tabela com o mesmo nome de outra tabela já existente no banco de dados;
- **DROP TABLE:** Esse comando apaga uma determinada tabela do banco de dados corrente.

Existem muitas outras declarações que fazem parte da DDL. Os mesmos comandos de alterar, criar e apagar, podem ser utilizados também para usuários do banco de dados, grupos e banco de dados. Essas declarações são poderosas, e podem mudar toda a característica do banco de dados, por isso elas são de acesso apenas de usuários com alto nível de permissões[Neves 2002][Neto 2003].

Da DML fazem parte declarações como, por exemplo:

- **SELECT:** Esse comando realiza a consulta ao banco de dados, retomando as tabelas, colunas, campos e registros especificados. Podem ser inseridos, em conjunto com esse comando, comandos de condição, agrupamento, ordenação, funções, dentre outros;
- **INSERT:** Esse comando insere novos registros na tabela especificada. Em seu uso, podemos citar, ou não, a ordem dos campos a serem inseridos;
- **DELETE:** Esse comando apaga registros de uma tabela especificada;
- **UPDATE:** Esse comando altera os dados existentes nos registros de uma tabela especificada. Pode adicionar novos dados, ou apenas modificar os já existentes.

Dentro de cada comando desses, o usuário pode fazer inúmeras combinações obter exatamente o que deseja. Para o funcionamento desses comandos é necessária a utilização de algumas palavras reservadas da linguagem, como, por exemplo, o FROM[Neves 2002][Neto 2003].

Capítulo 3

Proposta de Implementação

Como citado anteriormente no capítulo 1, o principal objetivo do trabalho consiste desenvolvimento de um sistema de monitoramento e armazenamento de dados coletados da planta LAMP, abrangendo diversos níveis hierárquicos presentes na pirâmide da automação industrial. Logo, serão apresentadas, neste capítulo, as implementações relativas a cada módulo do sistema, para que, em seguida, estas sejam relacionadas aos níveis já discutidos.

3.1 Problemática

Desenvolver um Sistema Supervisório capaz de capturar as variáveis disponíveis da Rede de Sensores *Foundation Fieldbus* do Laboratório A na Planta LAMP, e permitir a visualização, modificação de valores – quando couber – , e armazenamento em um banco de dados, para posterior consulta e uso.

O sistema deverá ser robusto o suficiente para armazenar todos os dados coletados pelos transmissores de temperatura, perceber quando os dados dos sensores de pressão deverão ser armazenados – momento em que ocorrem ensaio em campo –, gerar relatório dos dados armazenados para uso em outros sistemas, permitir configuração de tempo de amostragem de forma arbitrária, ter interface didática e intuitiva e oferecer informações úteis ao usuário.

3.1.1 Estrutura Disponível

A Rede de Sensores do Laboratório A na Planta LAMP, é composta pelos seguintes dispositivos:

- Três Sensores/Transmissores de Pressão Diferencial – LD302 –, ligados em série no duto que conecta o Tanque Misturador ao Tanque Tratador;

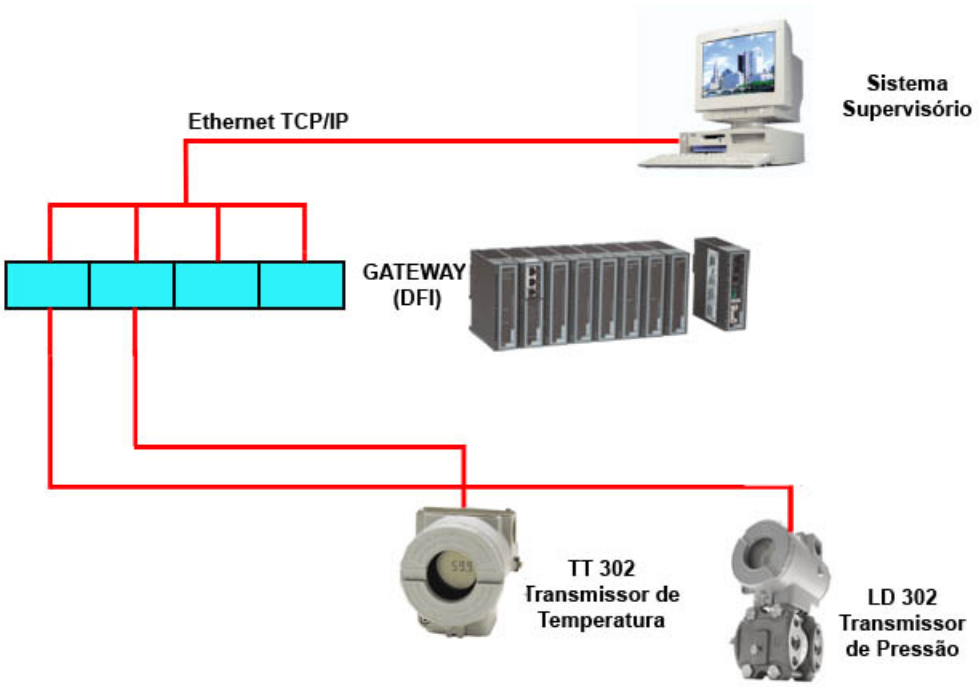


Figura 3.1: Ligação entre a rede e os instrumentos do Laboratório A

- Quatro Transmissores de Temperatura – TT302 – todos eles com elemento sensor PT 100 IEC de três fios, sendo que dois deles ligados ao Tanque de Água e outros dois ligados ao Tanque de Óleo;
- Uma DFI 302 (*Gateway*) modularizada:
  - *Backplane* DF1 – *Rack* com 4 *Slots*;
  - Fonte DF50 – *Power Supply for backplane* 90–264VAC;
  - Módulo Processador DF51 – DFI302 *Processor* 1x10 Mbps, 4xH1;
  - Fonte *FieldBus* DF52 – *Power Supply for Fieldbus*;
  - Impedância DF53 – *Power Supply Impedance for Fieldbus* (4 portas);
  - Terminador DF2 – Terminador para o último *rack*;
  - Cabo padrão *Ethernet* DF54 – *Twisted-Pair* (10 base T) *Cable* – Comprimento 2 m.

3.2 Projeto e Implementação do Banco de Dados

O projeto do banco de dados foi realizado no Software *DBDesigner (Data Base Designer*), para permitir uma visualização da estrutura a ser construída e em seguida imple-

mentado no SGBDR PostgresSQL.

Na figura 3.2 vê-se o conjunto de tabelas e relacionamentos previstos para existirem entre os dados.

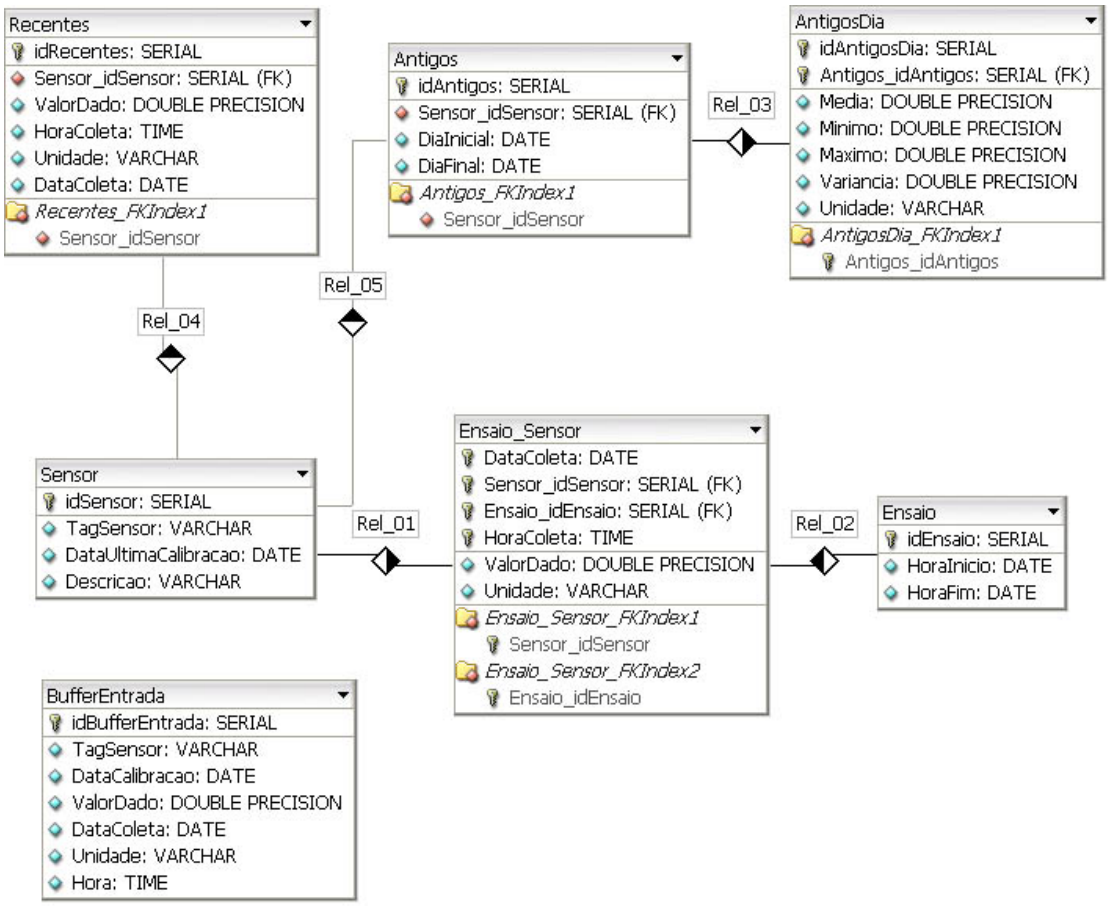


Figura 3.2: Estrutura projetada do Banco de Dados

A tabela BufferEntrada é utilizada para fazer a comunicação entre o programa supervisório e o banco de dados. O programa supervisório só enxerga essa tabela no banco, os dados vindos da planta são inseridos nessa tabela pelo programa. Os dados são distribuídos para as outras tabelas do banco através de triggers e procedures. Foi desenvolvida uma trigger no banco, que quando houvesse inserção na tabela BufferEntrada, esse dado, caso seja relevante, seja inserido na tabela Recentes, que é responsável por armazenar todos os dados vindos dos sensores em um determinado intervalo de tempo, definido como sendo 10 dias, mas que pode ser facilmente modificado. Essa trigger também verifica se o sensor já está cadastrado no banco, se não tiver, o banco o cadastra automaticamente, armazenando sua *tag*, que determinará que sensor seja, e a data da sua última calibração.

Caso um sensor seja retirado da rede, calibrado e colocado novamente na rede, o sistema o cadastra automaticamente como um ”novo sensor”, isso é útil porque a data de calibração é um parâmetro importante para avaliar a eficiência do sensor e de um possível algoritmo inteligente que esteja implementado no sensor.

Na tabela Recentes, também fica armazenado o dia e a hora da coleta do dado, para ser possível traçar gráficos em função do tempo dessa variável e fazer comparações.

As tabelas Antigos e AntigosDia são responsáveis por fazer uma espécie de ”compressão” dos dados. Quando passa o intervalo de tempo determinado uma *trigger* é responsável por transferir os dados da tabela Recentes para as tabelas Antigos e AntigosDia, a tabela AntigosDia guarda informações sobre cada dia dos sensores armazenados, e a tabela Antigos agrupa esses dias em períodos de tempo, com data inicial e data final. A tabela AntigosDia guarda atributos como média, mínimo, máximo e variância da variável.

Temos também as tabelas EnsaioSensor e Ensaio, na nossa planta existem ensaios, que acontecem em intervalos de tempo diferentes, e duram intervalos de tempo distintos também e os valores de pressão só são relevantes para serem armazenados se estiver ocorrendo um ensaio de separação. Quando ocorre ensaios, a pressão sai de um determinado intervalo, e usa-se essa premissa para determinar se está havendo ensaios ou não. Os valores de temperatura e pressão durante o ensaio são armazenados na tabela EnsaioSensor e a tabela Ensaio é responsável por armazenar os diferentes ensaios ocorridos com o mesmo sensor em momentos distintos. O local que verifica se os valores de pressão estão fora do intervalo normal é a *trigger* que transfere os dados da tabela de BufferEntrada para a tabela Recentes.

3.3 Criação do Sistema Supervisório

Definida a ferramenta de trabalho para o desenvolvimento do Sistema Supervisório – Elipse SCADA – foi levantado os trabalhos já relacionados feitos na área e no laboratório para que pudessem servir de base para desenvolvimento.

Um supervisório já existente da planta LAMP desenvolvido com a tecnologia Elipse SCADA, mas para outra rede de sensores, serviu como modelo. Na Figura 3.3 tem-se a tela principal do supervisório do Laboratório B – trabalha em pesquisas na área de medições de vazão, BSW e detecção de vazamento em dutos –, onde podemos visualizar *tags* de sensores nas quais a rede de industrial do Laboratório A – trabalha em pesquisas na área de algoritmos inteligentes em rede de sensores *Foundation Fieldbus* e desenvolvimento de *softwares* de gerencia de informações –, não incorpora.

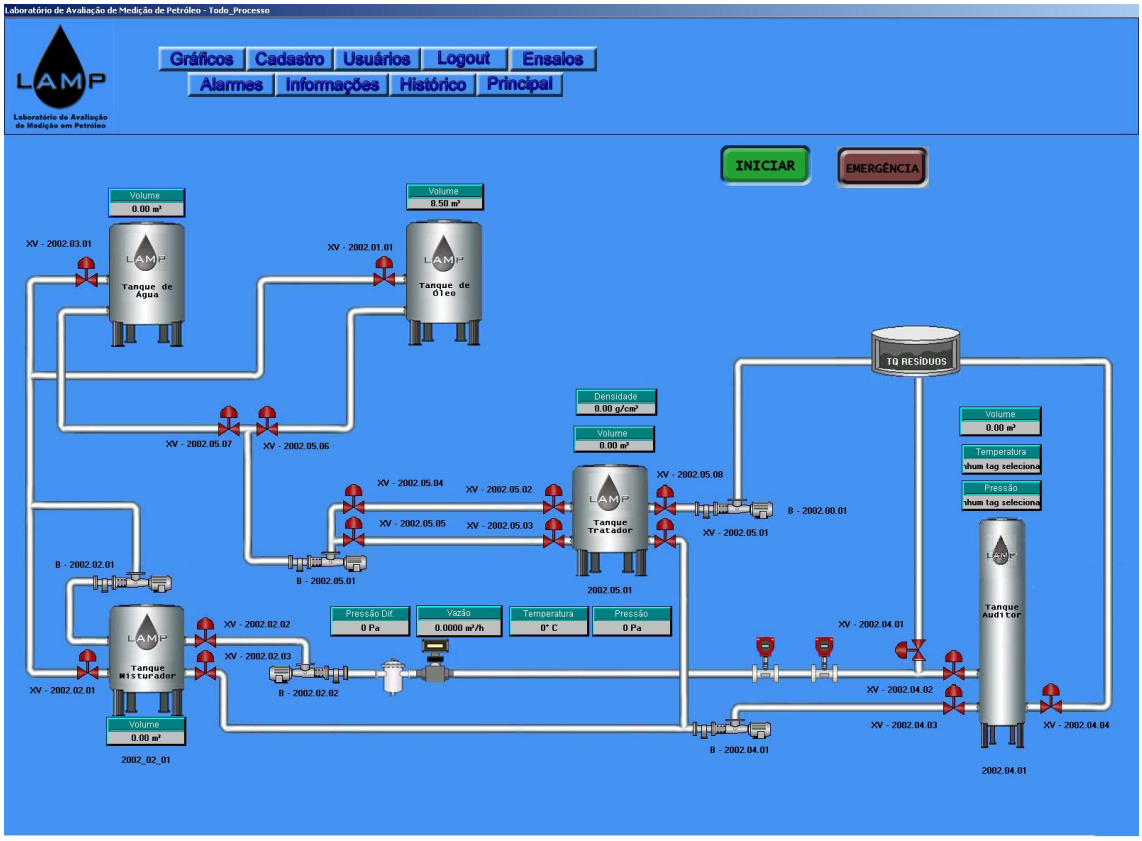


Figura 3.3: Tela principal do Supervisório – Laboratório B

3.3.1 Telas

O desenvolvimento das telas teve como princípio manter o padrão de telas já existente mas adicionando algumas características que promovessem uma interatividade, fazendo uso apenas dos componentes visuais *Bitmap*, *Button*, *Text*, e *Set Point*.

Na Tela Principal, figura 3.4, é ilustrado todo o processo capaz de ser monitorado pela rede de sensores do Laboratório A, visualizam-se componentes *Displays* que exibem informações sobre a variável monitorada. O título atribuído a cada *Tag* obedeceu ao número série do dispositivo, tal critério foi adotado pois a cada vez que adotamos uma nova configuração na rede *Foundation Fieldbus* o nome dos dispositivos podem mudar deixando assim confuso a identificação do sensor que realmente está monitorando as variáveis de temperatura de Água, Óleo e Pressão no duto.

No menu inferior da Tela Principal, se clicarmos no botão ”Intervalo de Amostragem”, é aberta uma a janela (Figura 3.5) onde é permitida a configuração de qual o intervalo desejado, em segundos, que ocorra o armazenamento dos dados coletados no banco de dados, o sistema só iniciará o armazenamento após o acionamento do botão ”Iniciar

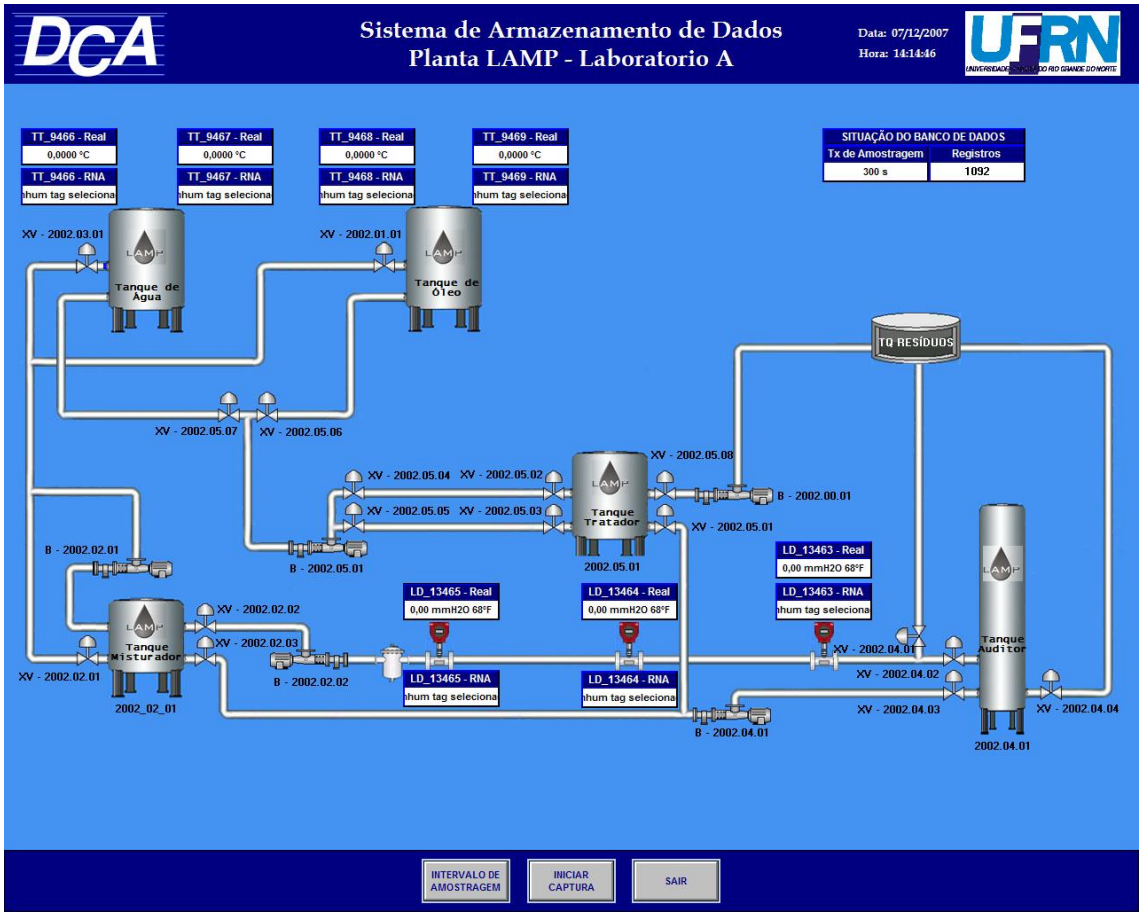


Figura 3.4: processo

Captura", e esta captura pode ser parada clicando novamente sobre o mesmo botão, agora identificado por "Finalizar Captura".

Clicando sobre cada um desses *Displays* ou sobre a imagem do dispositivo, no caso do sensor de pressão, uma tela com informações detalhadas e opção de geração de Relatórios, do sensor selecionado, será aberta (Figura 3.6).

Nesta tela é possível visualizar valores de alguns campos principais, e até mesmo setar novos valores para algumas *tags* que permitam escrita.

Também há a possibilidade de geração de um relatório com dados já coletados pelo sensor, obedecendo alguns padrões de formatação já definidos, assim como de extensão de arquivo. Após definidas as configurações do relatório, se clicarmos em "Gerar Relatório", no menu inferior, uma Tela solicita o local a ser salvo e qual o nome do arquivo.

Outra opção existente no menu inferior é "Descrição do Dispositivo" que mostra informações resumidas colhidas do manual do dispositivo, figuras3.7,3.8.

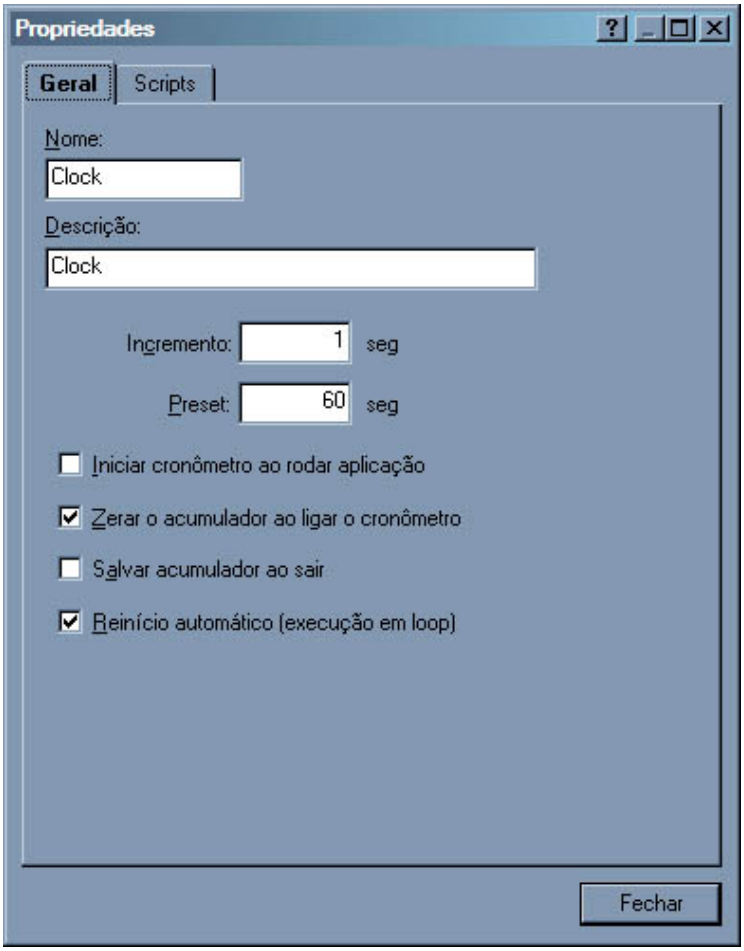


Figura 3.5: Configuração do Clock

3.3.2 Funcionamento em *Background*

O Elipse SCADA, possui uma linguagem de *Script* – linguagem de computador interpretada; linguagens de programação executadas em um interpretador – que permitem a execução de ações conforme algum procedimento ocorra.

Em cada componente inserido, uma aba *script*(Figura 3.9), permite a inserção desses comandos, obedecendo a uma sintaxe padronizada pela ferramenta.

Neste contexto, alguns *scripts* foram desenvolvidos buscando otimizar o uso dos recursos disponíveis. A seguir será descrito os *scripts* desenvolvido dentro de cada parte da Aplicação.

Aplicação – Principal

Ao ser iniciada a aplicação, um *script* carrega as últimas informações coletadas do sistema para que não ocorra erros na primeira inserção de novos registros no Banco de

Figura 3.6: Tela Detalhes

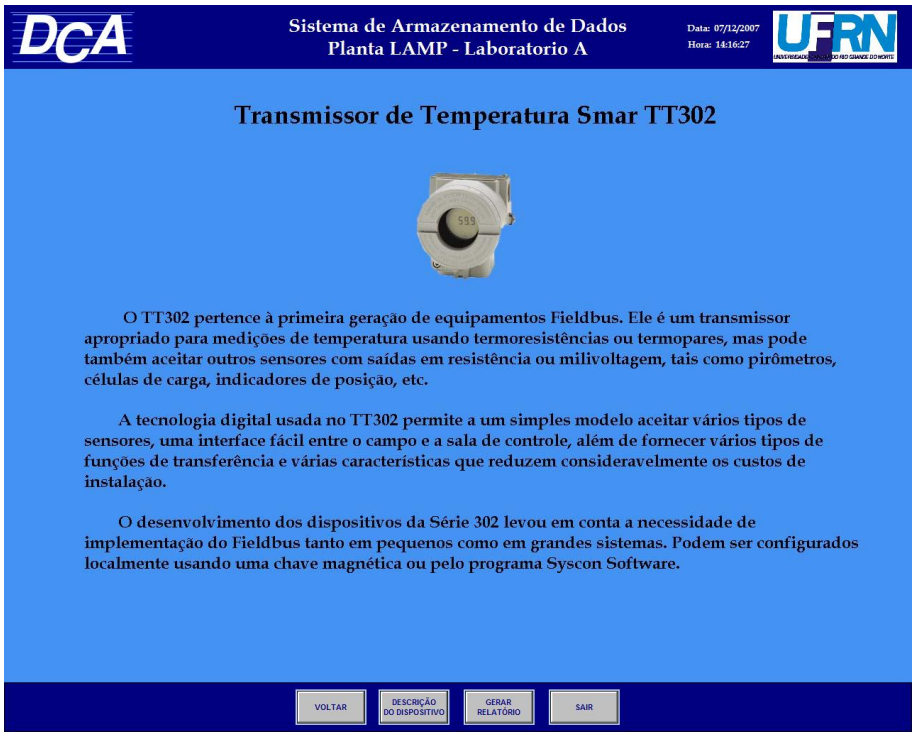

DCA
Sistema de Armazenamento de Dados
Planta LAMP - Laboratorio A
Data: 07/12/2007
Hora: 14:16:27
UFRN
Transmissor de Temperatura Smar TT302
O TT302 pertence à primeira geração de equipamentos Fieldbus. Ele é um transmissor apropriado para medições de temperatura usando termoresistências ou termopares, mas pode também aceitar outros sensores com saídas em resistência ou milivoltagem, tais como pirômetros, células de carga, indicadores de posição, etc.
A tecnologia digital usada no TT302 permite a um simples modelo aceitar vários tipos de sensores, uma interface fácil entre o campo e a sala de controle, além de fornecer vários tipos de funções de transferência e várias características que reduzem consideravelmente os custos de instalação.
O desenvolvimento dos dispositivos da Série 302 levou em conta a necessidade de implementação do Fieldbus tanto em pequenos como em grandes sistemas. Podem ser configurados localmente usando uma chave magnética ou pelo programa Syscon Software.
VOLTAR
DESCRIÇÃO DO DISPOSITIVO
GERAR RELATÓRIO
SAIR

Figura 3.7: Telas de Informações dos dispositivos – TT302

Dados. No mesmo instante o sistema configura em cada *Display*, da Tela Principal, as unidades matemáticas utilizadas nas configurações dos dispositivos de campo, atualiza

DCA
Sistema de Armazenamento de Dados
Planta LAMP - Laboratorio A
Data: 07/12/2007
Hora: 14:15:46
UFRN
Sensor/Transmissor de Pressão Diferencial Smar LD302
A linha de transmissores LD302 incorpora a altamente comprovada técnica de medição de pressão por célula capacitiva e a flexibilidade e precisão asseguradas pela utilização de microprocessador no seu circuito eletrônico. Ele pode ser utilizado para mecdoção de pressão diferencial, absoluta, manométrica , nível e vazão.
A tecnologia digital usada no LD302 permite a escolha de vários tipos de funções de transferência, um interfaceamento fácil entre o campo e a sala de controle e algumas características que reduzem consideravelmente a instalação, operação e os custos de manutenção.
O desenvolvimento dos dispositivos da Série 302 levou em conta a necessidade de implementação do Fieldbus tanto em pequenos como em grandes sistemas. Podem ser configurados localmente usando uma chave magnética ou pelo Syscon Software.
VOLTAR
DESCRIÇÃO DO DISPOSITIVO
GERAR RELATÓRIO
SAIR

Figura 3.8: Telas de Informações dos dispositivos – LD302

os valor do número de registros armazenados no Banco de Dados, e fica em espera de alguma interrupção solicitando o fechamento do sistema, seja ela a tecla ”*Esc*”, *Alt* + F4, ou botão ”Sair”, quando isso ocorre, ele salva em uma Receita(conjunto de valores pré-definidos que podem ser carregados para um grupo de *tags* a fim de configurar um processo específico) os últimos valores armazenados e finaliza todo o sistema.

Componente *Clock* – Relógio do Sistema

Este componente, não visual, foi inserido visando permitir, de forma independente, a configuração do intervalo de amostragem de dados a ser armazenado no Banco de Dados. Em seu *script* a cada vez que o contador estoura sua contagem ele executa o procedimento de capturar o valor atual de cada *tag* selecionada para armazenamento e envia para o banco. Este é o primeiro gargalo do sistema, pois o tempo que cada registro demora para ser confirmada a inserção é da ordem de décimos de segundo, tornando o sistema inoperante durante a realização deste *script*.

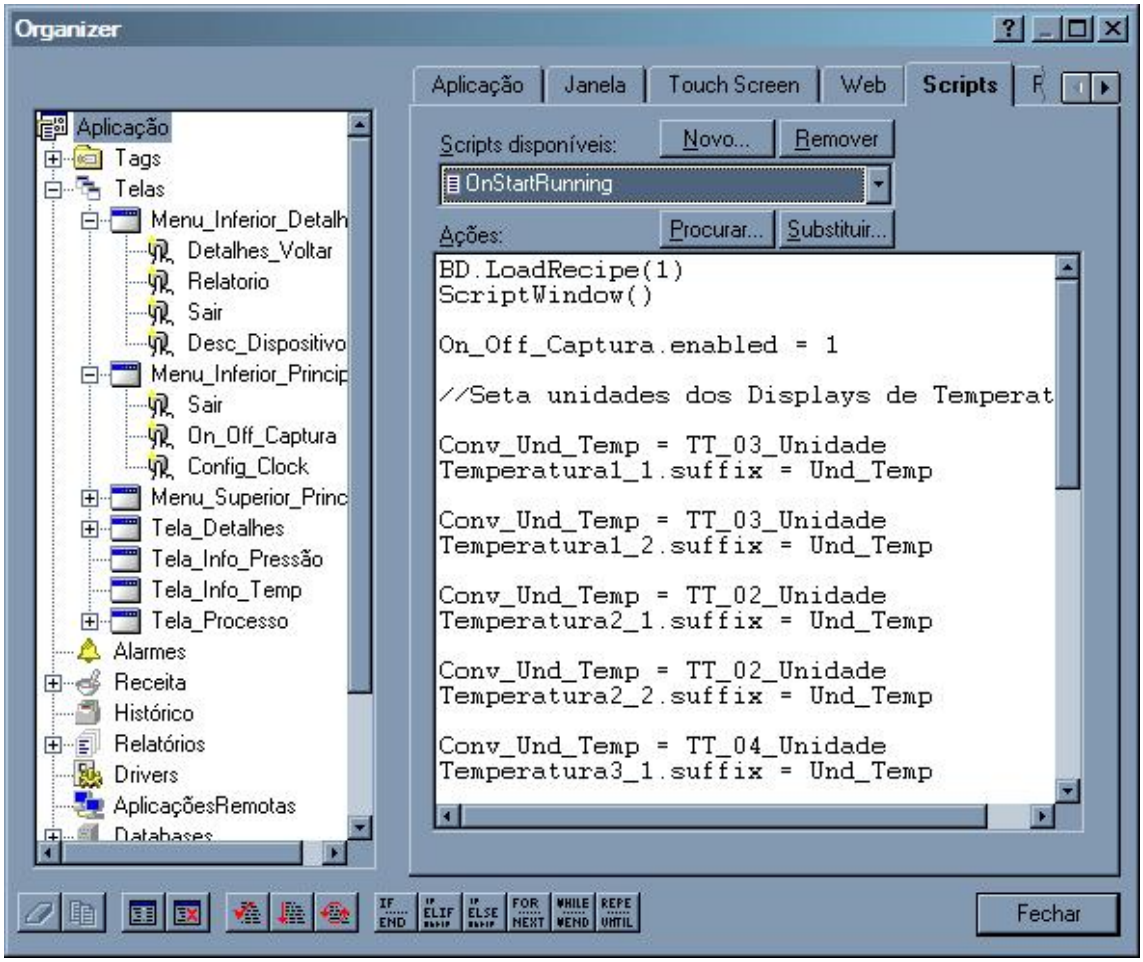


Figura 3.9: Janela de Edição de Scripts da Aplicação

Variáveis Locais

Como o Elipse SCADA não permite a criação de novas funções, criamos duas *tags* no próprio sistema e com elas fizemos a conversão de unidades utilizada nos sensores baseada na configuração do campo. Para esta conversão, de número inteiro para símbolo matemático, usamos a tabela de código de unidades dos blocos funcionais, como podemos ver nas tabelas 3.1 e 3.2.[Fieldbus 20 05]

Código (Inteiro)	Unidade de Tempreatura
1000	Kelvin
1001	Celsius
1002	Fahrenheit
1003	Rankine

Tabela 3.1: Tabela de Conversão Inteiro X Unidade de Temperatura.

Código (Inteiro)	Unidade de Pressão
1130	Pa
1132	Mpa
1133	kPa
1137	bar
1138	mbar
1139	torr
1140	atm
1141	psi
1144	gcm2
1145	kgfcm2
1147	inH2O 4°C
1148	inH2O 68°F
1150	mmH2O 4 °C
1151	mmH2O 68 °F
1154	ftH2O 68 °F

Tabela 3.2: Tabela de Conversão Inteiro X Unidade de Pressão.

Botão ”Gerar Relatório”

Em seu *script* uma variável temporária para armazenamento da *string* do caminho de salvamento do arquivo é criada, e o resultado de uma consulta ao banco é armazenada no arquivo selecionado pelo usuário. Aqui encontramos o segundo gargalo do sistema, a geração do relatório, pois após realizada a consulta devemos percorrer registro a registro e concatenando-os para posteriormente serem armazenados no arquivo, isto pode levar cerca de alguns segundos, podendo até travar o sistema caso existam muitos registros no banco referentes a tal sensor.

3.3.3 Comunicação Supervisório X Banco de Dados

Criação de Fonte de Dados ODBC

Para que houvesse a comunicação Supervisório X Banco de Dados, se faz necessária a criação de um *driver* ODBC, sendo criado da seguinte forma: A partir do Painel de Controle do MS Windows XP, que pode ser acessado através da opção Configurações do Menu Iniciar do Windows, escolhe-se Ferramentas Administrativas e depois Fontes de Dados ODBC(Figura 3.10); Na aba Fonte de Dados de Sistema, opta-se por Adicionar uma nova fonte; Selecionando o *driver* PostgreSQL ANSI – por ser o padrão de *driver* re-

conhecido pelo Software Elipse; Preenche-se os campos respectivos da tela (Figura 3.11) com os dados necessário e assim é finalizada a criação da conexão ODBC.

Figura 3.10: Administrador de Fonte de Dados do MS Windows XP

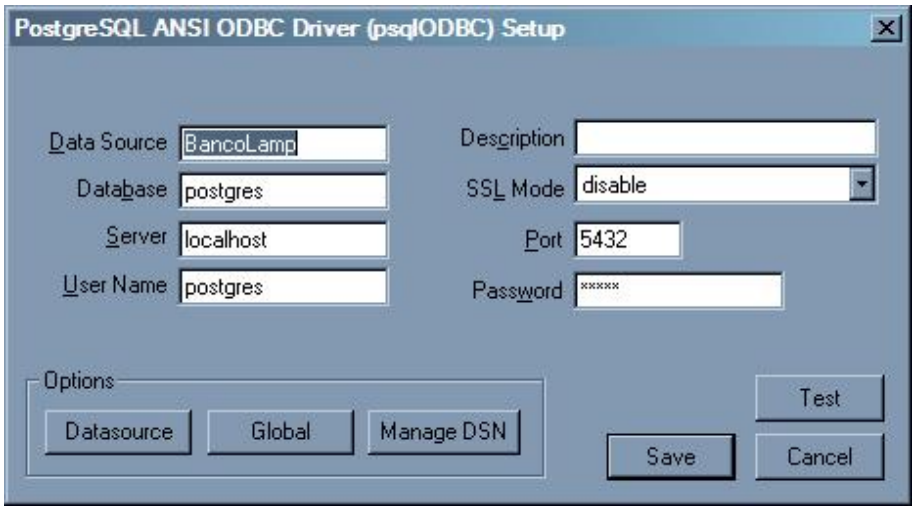


Figura 3.11: Janela de configuração da nova fonte de dados PostgreSQL ANSI ODBC Driver criada

3.4 Validação do Sistema

Para o desenvolvimento de um Projeto Funcional de uma rede *Foundation Fieldbus* deve-se seguir uma seqüência de passos, a qual se inicia com a criação de um esquema lógico e em seguida sua implementação.

Um esquema lógico é definido o planejamento de como serão utilizados os recursos disponíveis em sua rede industrial, dispositivos físicos, sensores e atuadores, e o modo de configurá-los para atingir o objetivo pretendido.

Na implementação há a alocação, onde nós unimos as duas bases do esquema lógico, ou seja, colocamos dentro dos sensores e atuadores, a configuração projetada.

Para validar o sistema, levantamos primeiramente os dispositivos disponíveis na rede, como visto na seção 3.1, e projetamos uma configuração simples na qual disponibilizaria na rede a variável analógica monitorada por cada sensor, além das informações sempre existentes em cada dispositivo inserido.

3.4.1 Configuração de Rede *Foundation Fieldbus*

Através do *Software Syscon* que faz parte do Pacote *System302* da SMAR é possivel realizar a configuração de uma Rede *Foundation Fieldbus* . Para tal deve-se inicialmente realizar a conexão com a DFI302 e identificar quais instrumentos estão associados aos canais da mesma.

No caso da Planta LAMP trabalhamos com dois canais da DFI302, sendo no primeiro inserida a rede de sensores transmissores de pressão e no canal dois a rede de transmissores de temperatura, esta configuração pode ser vista na figura 3.12.

Em seguida associamos cada dispositivo a configuração do canal correspondente, e em cada um deles instânciamos e parametrizamos cada bloco funcional.

Pode-se observar abaixo a parametrização de alguns blocos disponíveis para os dispositivos supra citados.

Em um Bloco Funcional de Diagnóstico deve ser configurado obrigatoriamente o parâmetro TARGET do grupo MODE_BLOCK, em nosso caso configuramos como AUTO, indicando que o funcionamento do bloco será feito maneira automática.

Em um Bloco Funcional *Tranducer* (Transdutor) deve ser configurado obrigatoriamente o parâmetro TARGET do grupo MODE_BLOCK, em nosso caso configuramos como AUTO, indicando que o será realizada a conversão da variável para um padrão interpretável na rede.

Em um Bloco Funcional AI (*Analog Input*) deve ser configurado obrigatoriamente os

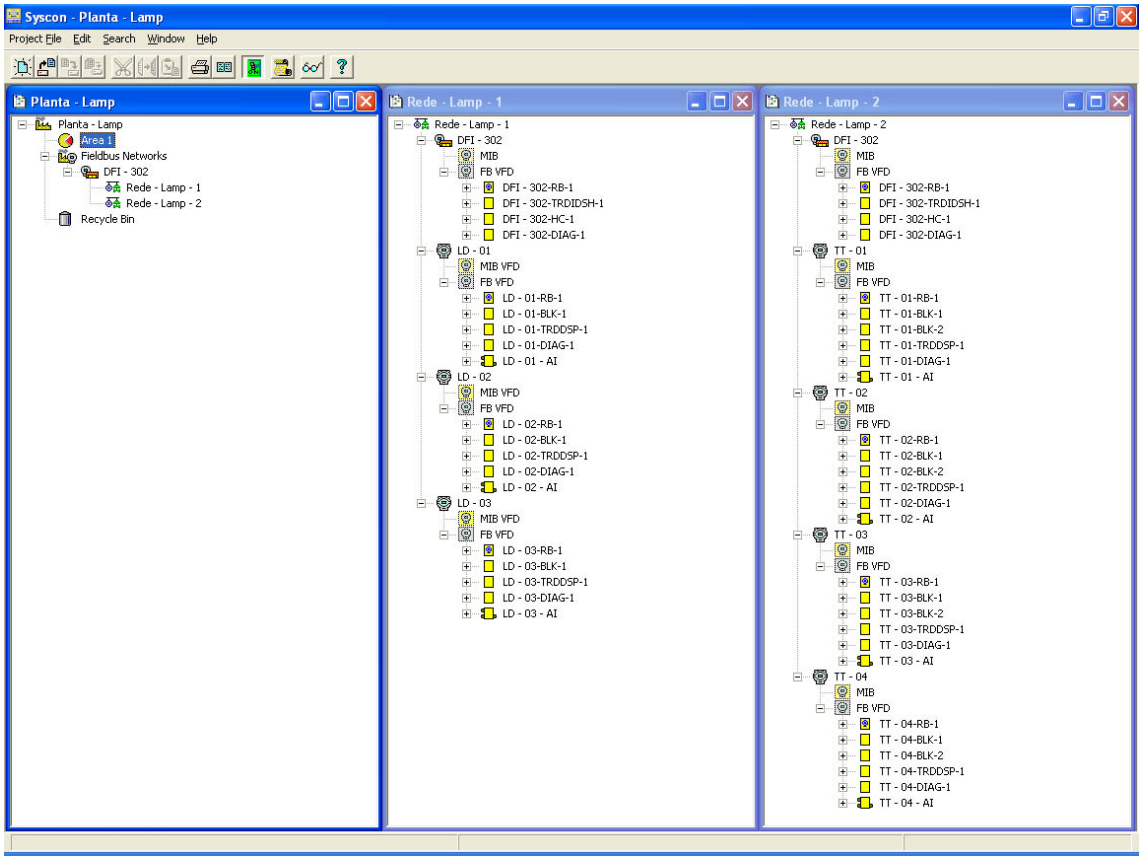


Figura 3.12: Config Syscon

parâmetros:

- TARGET do grupo MODE_BLOCK, em nosso caso configuramos como AUTO;
- L_Type como DIRECT, o qual repassará para a rede o padrão da conversão realizada pelo *TRANSDUCER*;
- CHANNEL que indicará qual canal do sensor será utilizado.

Em um Bloco Funcional *Resource* (Recursos) deve ser configurado obrigatoriamente o parâmetro TARGET do grupo MODE_BLOCK, em nosso caso configuramos como AUTO, apenas para indicar de maneira automática características do dispositivo.

Após a configuração de cada dispositivo é necessária a Ativação da Configuração, onde será criado o Servidor OPC, denominado Smar.DFIOLEServer.0, e o mapeamento de cada configuração criada será feito.

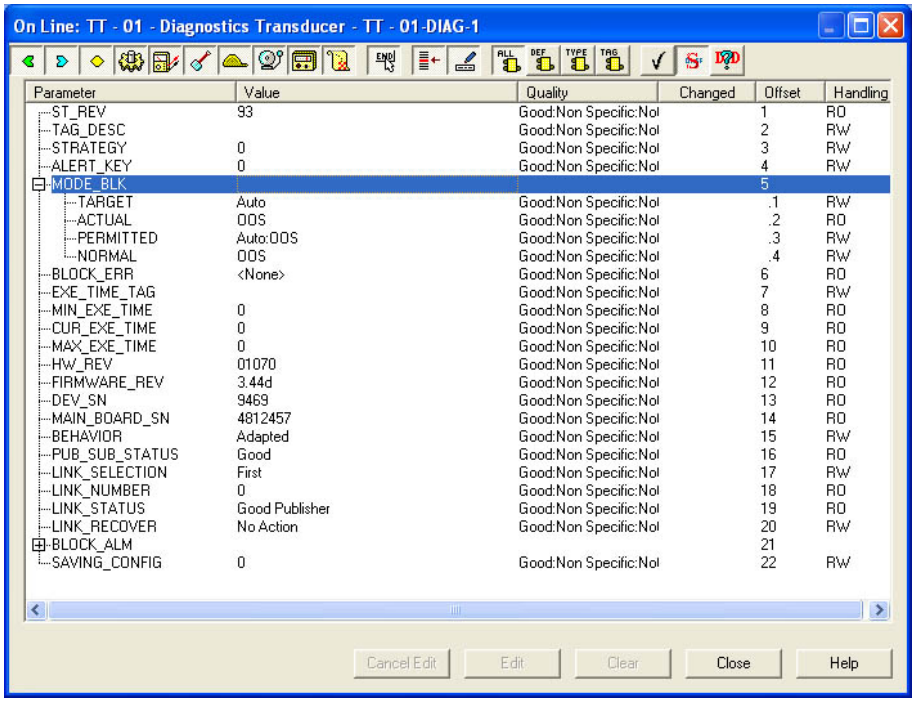

On Line: TT - 01 - Diagnostics Transducer - TT - 01-DIAG-1
Parameter
Value
Quality
Changed
Offset
Handling
ST_REV 93 Good:Non Specific:Not Limited 1 RO
TAG_DESC Good:Non Specific:Not Limited 2 RW
STRATEGY 0 Good:Non Specific:Not Limited 3 RW
ALERT_KEY 0 Good:Non Specific:Not Limited 4 RW
MODE_BLK 5
TARGET Auto Good:Non Specific:Not Limited .1 RW
ACTUAL OOS Good:Non Specific:Not Limited .2 RO
PERMITTED Auto:OOS Good:Non Specific:Not Limited .3 RW
NORMAL OOS Good:Non Specific:Not Limited .4 RW
BLOCK_ERR <None> Good:Non Specific:Not Limited 6 RO
EXE_TIME_TAG Good:Non Specific:Not Limited 7 RW
MIN_EXE_TIME 0 Good:Non Specific:Not Limited 8 RO
CUR_EXE_TIME 0 Good:Non Specific:Not Limited 9 RO
MAX_EXE_TIME 0 Good:Non Specific:Not Limited 10 RO
HW_REV 01070 Good:Non Specific:Not Limited 11 RO
FIRMWARE_REV 3.44d Good:Non Specific:Not Limited 12 RO
DEV_SN 9469 Good:Non Specific:Not Limited 13 RO
MAIN_BOARD_SN 4812457 Good:Non Specific:Not Limited 14 RO
BEHAVIOR Adapted Good:Non Specific:Not Limited 15 RW
PUB_SUB_STATUS Good Good:Non Specific:Not Limited 16 RO
LINK_SELECTION First Good:Non Specific:Not Limited 17 RW
LINK_NUMBER 0 Good:Non Specific:Not Limited 18 RO
LINK_STATUS Good Publisher Good:Non Specific:Not Limited 19 RO
LINK_RECOVER No Action Good:Non Specific:Not Limited 20 RW
BLOCK_ALM 21
SAVING_CONFIG 0 Good:Non Specific:Not Limited 22 RW
Cancel Edit
Edit
Clear
Close
Help

Figura 3.13: Bloco Funcional de Diagnóstico

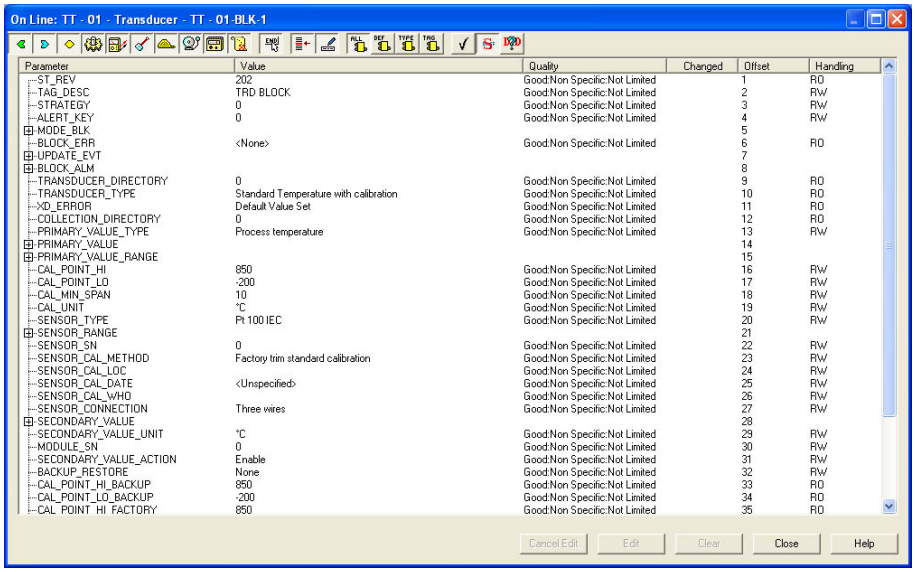

On Line: TT - 01 - Transducer - TT - 01-BLK-1
Parameter
Value
Quality
Changed
Offset
Handling
ST_REV 202 Good:Non Specific:Not Limited 1 RO
TAG_DESC TRD BLOCK Good:Non Specific:Not Limited 2 RW
STRATEGY 0 Good:Non Specific:Not Limited 3 RW
ALERT_KEY 0 Good:Non Specific:Not Limited 4 RW
MODE_BLK 5
BLOCK_ERR <None> Good:Non Specific:Not Limited 6 RO
UPDATE_EVT 7
BLOCK_ALM 8
TRANSDUCER_DIRECTORY 0 Good:Non Specific:Not Limited 9 RO
TRANSDUCER_TYPE Standard Temperature with calibration Good:Non Specific:Not Limited 10 RO
XD_ERROR Default Value Set Good:Non Specific:Not Limited 11 RO
COLLECTION_DIRECTORY 0 Good:Non Specific:Not Limited 12 RO
PRIMARY_VALUE_TYPE Process temperature Good:Non Specific:Not Limited 13 RW
PRIMARY_VALUE 14
PRIMARY_VALUE_RANGE 15
CAL_POINT_HI 850 Good:Non Specific:Not Limited 16 RW
CAL_POINT_LO -200 Good:Non Specific:Not Limited 17 RW
CAL_MIN_SPAN 10 Good:Non Specific:Not Limited 18 RW
CAL_UNIT °C Good:Non Specific:Not Limited 19 RW
SENSOR_TYPE Pt 100 IEC Good:Non Specific:Not Limited 20 RW
SENSOR_RANGE 21
SENSOR_SN 0 Good:Non Specific:Not Limited 22 RW
SENSOR_CAL_METHOD Factory trim standard calibration Good:Non Specific:Not Limited 23 RW
SENSOR_CAL_LOC Good:Non Specific:Not Limited 24 RW
SENSOR_CAL_DATE <Unspecified> Good:Non Specific:Not Limited 25 RW
SENSOR_CAL_WHO Good:Non Specific:Not Limited 26 RW
SENSOR_CONNECTION Three wires Good:Non Specific:Not Limited 27 RW
SECONDARY_VALUE 28
SECONDARY_VALUE_UNIT °C Good:Non Specific:Not Limited 29 RW
MODULE_SN 0 Good:Non Specific:Not Limited 30 RW
SECONDARY_VALUE_ACTION Enable Good:Non Specific:Not Limited 31 RW
BACKUP_RESTORE None Good:Non Specific:Not Limited 32 RW
CAL_POINT_HI_BACKUP 850 Good:Non Specific:Not Limited 33 RO
CAL_POINT_LO_BACKUP -200 Good:Non Specific:Not Limited 34 RO
CAL_POINT_HI_FACTORY 850 Good:Non Specific:Not Limited 35 RO
Cancel Edit
Edit
Clear
Close
Help

Figura 3.14: Bloco Funcional Transdutor

3.4.2 Comunicação rede *Foundation Fieldbus* X Supervisório

No Elipse Scada, inseriu-se um objeto OPCServer – um cliente OPC que possibilita a comunicação com um determinado equipamento ou dispositivo, utilizando o protocolo OPC – o que permitiu o envio e recebimento de dados de *tags* em tempo real, a comuni-

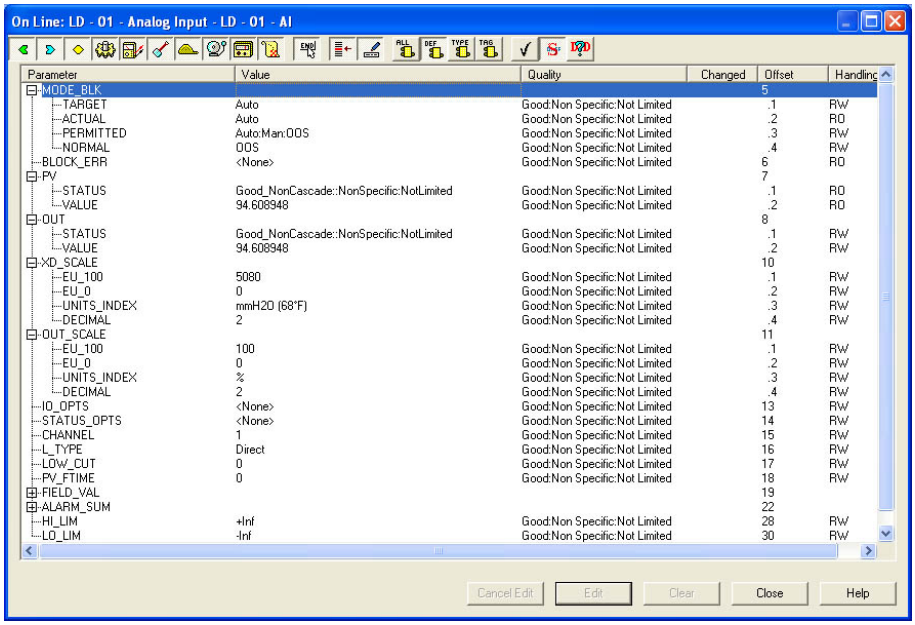


Figura 3.15: Bloco Funcional AI(Entrada Analógica)

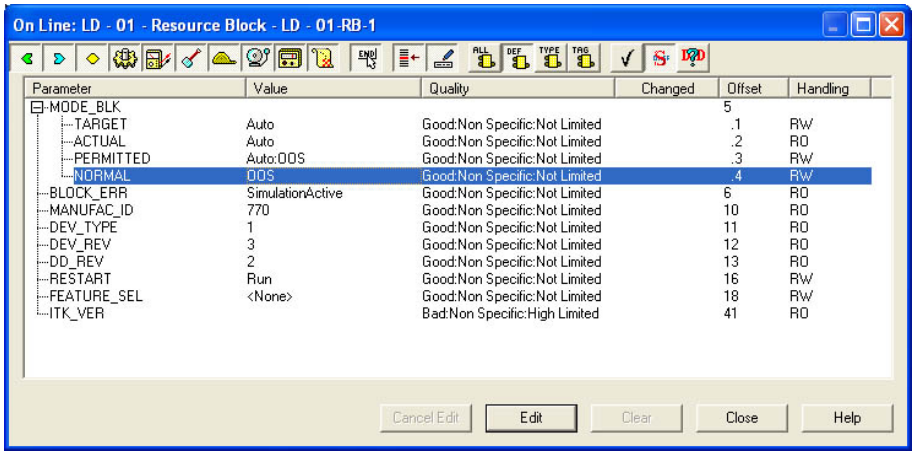


Figura 3.16: Bloco Funcional de Recursos

cação ocorre com o servidor OPC configurado no computador que que instância a lógica do sistema, sendo que este captura as *tags* disponíveis pela DFI 302.

Na figura 3.17 visualizamos o cliente OPC criado. Para sua configuração inicialmente localiza-se um servidor, no caso deste trabalho Smar.DFIOLEServer.0, em seguida importamos as *tags* disponíveis e relevantes para o sistema. Aqui importamos as seguintes *tags*:

- DIAG-1.DEV_SN – o número serial do dispositivo acessado;
- BLK-1.PRIMARY_VALUE_RANGE.EU_100 – limite máximo possível de ser me-

dido pelo sensor, sem danificá-lo;

- BLK-1.PRIMARY_VALUE_RANGE.EU_0 – limite mínimo possível de ser medido pelo sensor, sem danificá-lo;
- BLK-1.PRIMARY_VALUE.VALUE – valor atual da variável medida;
- BLK-1.PRIMARY_VALUE_RANGE.UNITS_INDEX – unidade utilizada na para quantificar o valor da variável medida.

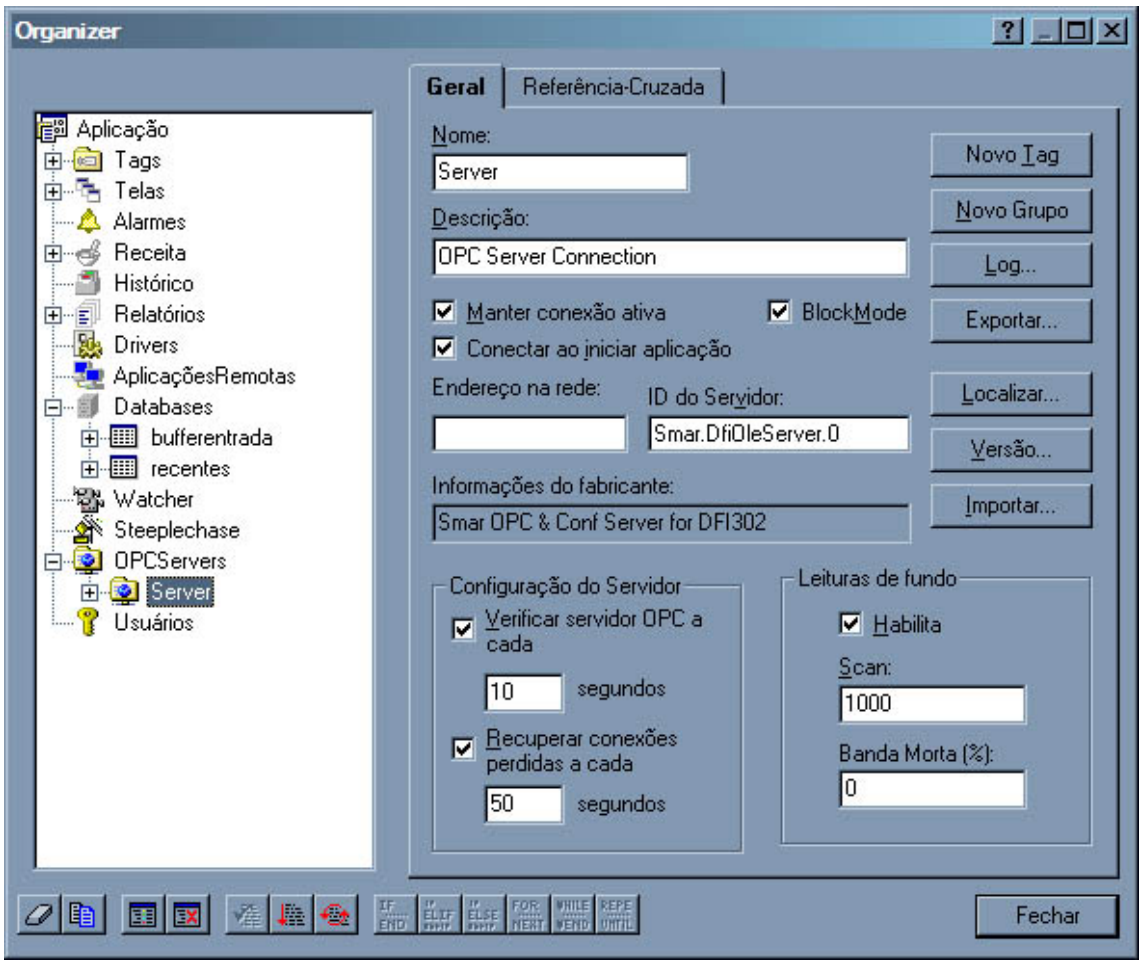
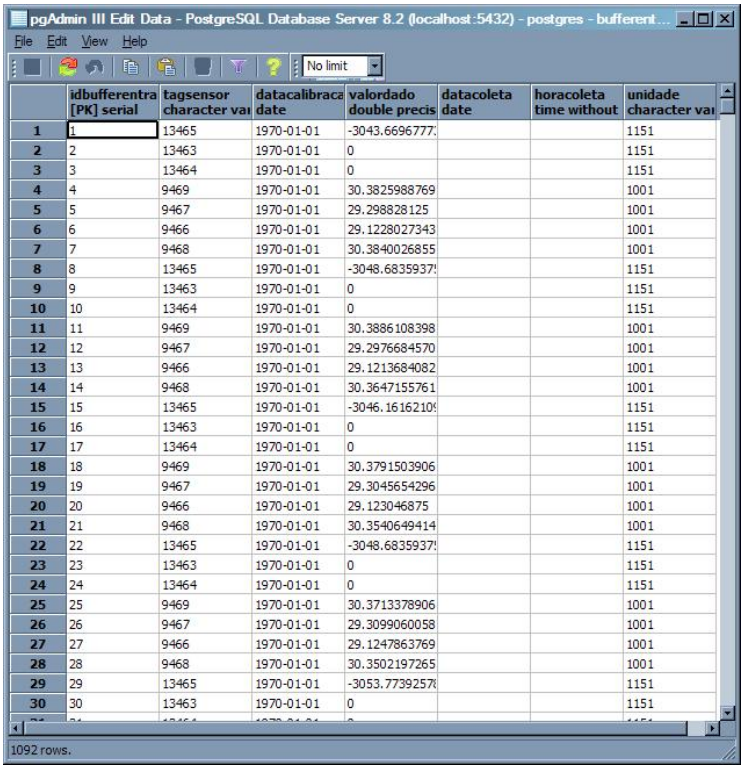


Figura 3.17: Cliente OPC do Elipse SCADA

Cada uma dessas *tags* serão manipuladas de forma transparente ao usuário como já explicado na seção 3.3.2, fechando assim a comunicação chão de fábrica sistema supervisório.

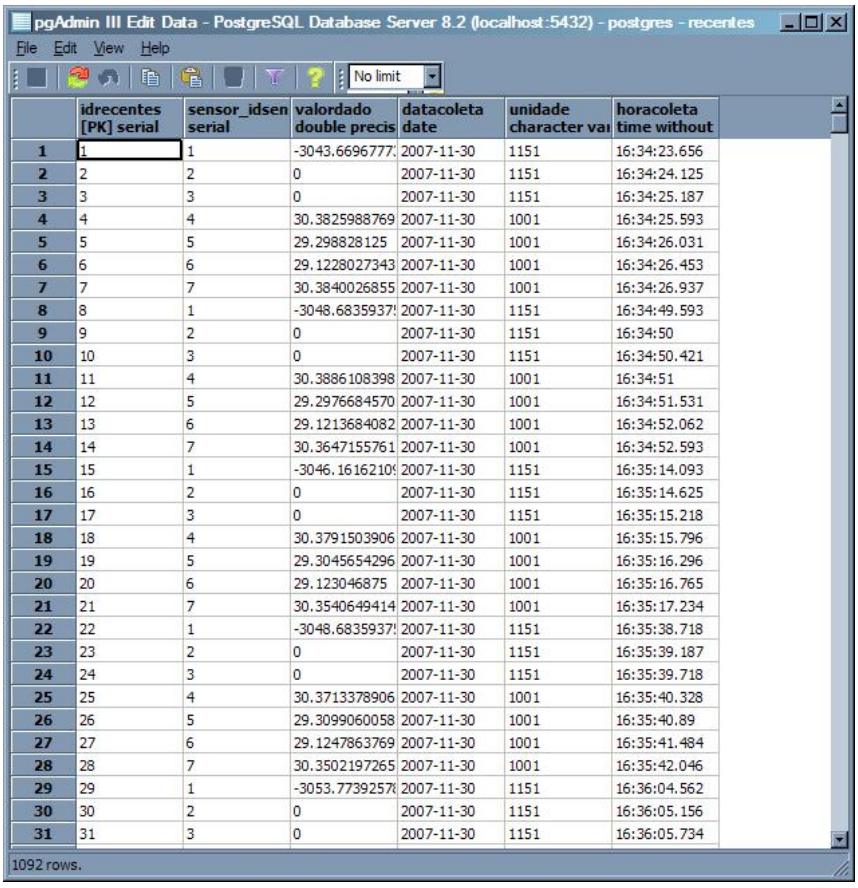
3.4.3 Dados Inseridos no Banco

Nas imagens 3.18,3.19,3.20 visualiza-se parte dos dados que foram armazenados a partir dos ensaio de testes realizados.


pgAdmin III Edit Data - PostgreSQL Database Server 8.2 (localhost:5432) - postgres - bufferent...
File Edit View Help
No limit
1092 rows.

	idbufferentra [PK] serial	tagsensor character var	datacalibraca date	valordado double precis	datacoleta date	horacoleta time without	unidade character var
1	1	13465	1970-01-01	-3043.6696777			1151
2	2	13463	1970-01-01	0			1151
3	3	13464	1970-01-01	0			1151
4	4	9469	1970-01-01	30.3825988769			1001
5	5	9467	1970-01-01	29.298828125			1001
6	6	9466	1970-01-01	29.1228027343			1001
7	7	9468	1970-01-01	30.3840026855			1001
8	8	13465	1970-01-01	-3048.6835937			1151
9	9	13463	1970-01-01	0			1151
10	10	13464	1970-01-01	0			1151
11	11	9469	1970-01-01	30.3886108398			1001
12	12	9467	1970-01-01	29.2976684570			1001
13	13	9466	1970-01-01	29.1213684082			1001
14	14	9468	1970-01-01	30.3647155761			1001
15	15	13465	1970-01-01	-3046.1616210			1151
16	16	13463	1970-01-01	0			1151
17	17	13464	1970-01-01	0			1151
18	18	9469	1970-01-01	30.3791503906			1001
19	19	9467	1970-01-01	29.3045654296			1001
20	20	9466	1970-01-01	29.123046875			1001
21	21	9468	1970-01-01	30.3540649414			1001
22	22	13465	1970-01-01	-3048.6835937			1151
23	23	13463	1970-01-01	0			1151
24	24	13464	1970-01-01	0			1151
25	25	9469	1970-01-01	30.3713378906			1001
26	26	9467	1970-01-01	29.3099060058			1001
27	27	9466	1970-01-01	29.1247863769			1001
28	28	9468	1970-01-01	30.3502197265			1001
29	29	13465	1970-01-01	-3053.7739257			1151
30	30	13463	1970-01-01	0			1151

Figura 3.18: Tabela BufferEntrada



pgAdmin III Edit Data - PostgreSQL Database Server 8.2 (localhost:5432) - postgres - recentes

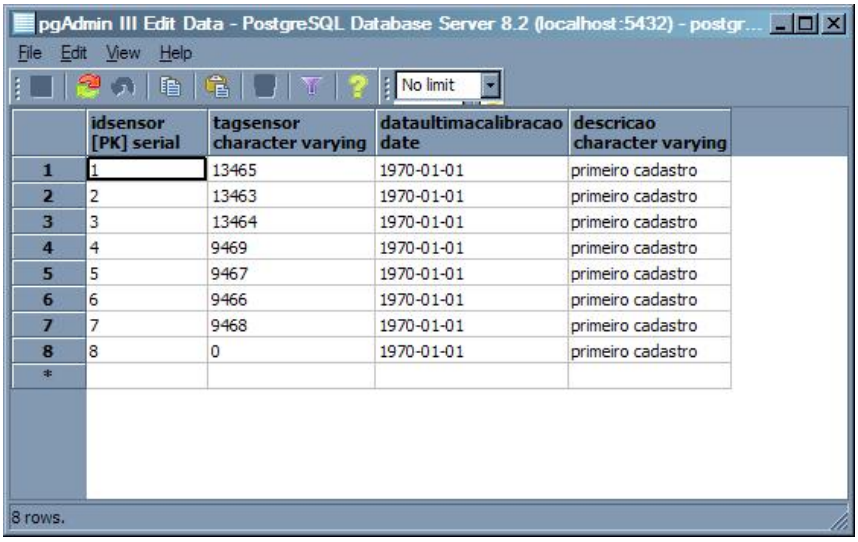
File Edit View Help

No limit

	idrecentes [PK] serial	sensor_idsen serial	valordado double precis	datacoleta date	unidade character var	horacoleta time without
1	1	1	-3043.6696777	2007-11-30	1151	16:34:23.656
2	2	2	0	2007-11-30	1151	16:34:24.125
3	3	3	0	2007-11-30	1151	16:34:25.187
4	4	4	30.3825988769	2007-11-30	1001	16:34:25.593
5	5	5	29.298828125	2007-11-30	1001	16:34:26.031
6	6	6	29.1228027343	2007-11-30	1001	16:34:26.453
7	7	7	30.3840026855	2007-11-30	1001	16:34:26.937
8	8	1	-3048.6835937	2007-11-30	1151	16:34:49.593
9	9	2	0	2007-11-30	1151	16:34:50
10	10	3	0	2007-11-30	1151	16:34:50.421
11	11	4	30.3886108398	2007-11-30	1001	16:34:51
12	12	5	29.2976684570	2007-11-30	1001	16:34:51.531
13	13	6	29.1213684082	2007-11-30	1001	16:34:52.062
14	14	7	30.3647155761	2007-11-30	1001	16:34:52.593
15	15	1	-3046.1616210	2007-11-30	1151	16:35:14.093
16	16	2	0	2007-11-30	1151	16:35:14.625
17	17	3	0	2007-11-30	1151	16:35:15.218
18	18	4	30.3791503906	2007-11-30	1001	16:35:15.796
19	19	5	29.3045654296	2007-11-30	1001	16:35:16.296
20	20	6	29.123046875	2007-11-30	1001	16:35:16.765
21	21	7	30.3540649414	2007-11-30	1001	16:35:17.234
22	22	1	-3048.6835937	2007-11-30	1151	16:35:38.718
23	23	2	0	2007-11-30	1151	16:35:39.187
24	24	3	0	2007-11-30	1151	16:35:39.718
25	25	4	30.3713378906	2007-11-30	1001	16:35:40.328
26	26	5	29.3099060058	2007-11-30	1001	16:35:40.89
27	27	6	29.1247863769	2007-11-30	1001	16:35:41.484
28	28	7	30.3502197265	2007-11-30	1001	16:35:42.046
29	29	1	-3053.7739257	2007-11-30	1151	16:36:04.562
30	30	2	0	2007-11-30	1151	16:36:05.156
31	31	3	0	2007-11-30	1151	16:36:05.734

1092 rows.

Figura 3.19: Tabela Recentes



pgAdmin III Edit Data - PostgreSQL Database Server 8.2 (localhost:5432) - postgr...

File Edit View Help

No limit

	idsensor [PK] serial	tagsensor character varying	dataultimacalibracao date	descricao character varying
1	1	13465	1970-01-01	primeiro cadastro
2	2	13463	1970-01-01	primeiro cadastro
3	3	13464	1970-01-01	primeiro cadastro
4	4	9469	1970-01-01	primeiro cadastro
5	5	9467	1970-01-01	primeiro cadastro
6	6	9466	1970-01-01	primeiro cadastro
7	7	9468	1970-01-01	primeiro cadastro
8	8	0	1970-01-01	primeiro cadastro
*				

8 rows.

Figura 3.20: Tabela Sensores

Capítulo 4

Conclusões

O trabalho desenvolvido permitiu uma interação direta com três níveis da pirâmide da Autmomação, sendo eles: Nível de sensores e atuadores, Controle Supervisório e Gerenciamento. Já os demais níveis, Controle Regulatório, Alarme e Intertravamento, foram estudados, apesar de não terem sido utilizados diretamente no trabalho.

Este trabalho permitiu a percepção de que o uso do software Supervisório Elipse SCADA, representa uma boa opção para o monitoramento das variáveis do processo, comunicação via OPC, Geração de Históricos e Alarmes, entretanto no ponto armazenamento de dados em um BDR (Banco de Dados Relacional) deixa a desejar em velocidade, por não ter a robustez suficiente – funções mais rápidas de Inserção, Remoção, Consultas, Captura de dados armazenados, entre outras – para trabalhar em conjunto com um BDR. Essa crítica foi levada ao conhecimento dos desenvolvedores do Software durante o desenvolvimento deste trabalho, via contatos ocorridos com o Suporte Técnico do Software Elipse SCADA, e obteve-se a resposta de que realmente há esta deficiência, e um outro software também desenvolvido pela Elipse, o Elipse E3, possui a robustez e o desempenho desejado no armazenamento e manipulação de dados, além de possuir um Banco de Dados próprio incorporado ao sistema.

Considerando a falta de velocidade dos resultados na manipulação de dados junto ao BDR, o supervisório desenvolvido permitirá validar trabalhos como [Silva 2005], [Lima 2004], entre outros futuros, que fazem uso de qualidades intrínsecas a dispositivos *Foundation Fieldbus* podendo ratificar tais teses.

Outro ganho para comunidade acadêmica será o banco de informações comportamentais de sensores no chão de fábrica, que ficará acessível para futuros projetos na área de controle, previsão de séries temporais, auto-calibração de dispositivos sensores, inferência de outras grandezas, estudo de padrões de falhas, entre outras aplicações.

Pode-se dizer então para que se tenha um desempenho satisfatório deve-se criar um outro sistema que gere relatórios dos dados armazenados, e no armazenamento de dados

criar um armazenamento de dados temporário em arquivos e posteriormente repassar para o banco de dados em momentos de finalização do sistema, ou dentro de um intervalo de tempo maior.

Capítulo 5

Cronograma de Execução

1. Integração do software supervisório do laboratório, com o sistema desenvolvido;
2. Estudo e projeto do software de gerar relatórios;
3. Instalação da DFI e dos sensores da planta;
4. Instanciação de estratégias de controle nos sensores *Foundation Fieldbus* da rede;
5. Coleta de dados dos sensores;
6. Análise de resultados;
7. Elaboração da monografia.

Atividades	3° Trimestre - 2007	4° Trimestre - 2007	1° Trimestre - 2008
1	**X**	**X**	
2	**X**	**X**	
3	**X**	**X**	
4		**X**	
5		**X**	
6		**X**	
7		**X**	**X**

Referências Bibliográficas

Elipse, Software (2007), Elipse scada - hmi/scada software - manual do usuário. Manual do Usuário.

Fieldbus, Foundation (20 05), Manual de instruções dos blocos funcionais *Foundation Fieldbus*.

Filho, Constantino Seixas (2000), A automação nos anos 2000 - uma análise das novas fronteiras da automação, Relatório técnico, ATAN Sistemas de Automação.

LAMP (2003), Manual teórico - lamp - laboratório de avaliação de medição em petróleo. REde 10/01.

Lima, Fábio S. (2004), Estratégia de escalonamento de controladores pid baseado em regras fuzzy para redes industriais foundation fieldbus usando blocos padrões, Dissertação de mestrado, Universidade Federal do Rio Grande do Norte.

Maitelli, André Laurindo (2003), Controladores lógicos programáveis. Notas de aula da disciplina "Controladores Lógicos Programáveis".

Neto, Alvaro Pereira (2003), Postgresql, técnicas avançadas versão open source 7.x.

Neves, Denise Lemes Fernandes (2002), Postgresql, conceitos e aplicações.

Silva, Ana Paula G. e Marcelo Salvador (2005), O que são sistemas supervisórios? http://www.elipse.com.br/. Visitada em Novembro de 2007.

Silva, Diego Rodrigo Cabral (2005), Redes neurais artificiais no ambiente de redes industriais *Foundation Fieldbus* usando blocos funcionais padrões, Dissertação de mestrado, Universidade Federal do Rio Grande do Norte.

Silveira, Leonardo e Weldson Q. Lima (2003), Um breve histórico conceitual da automação industrial e redes para automação industrial. Redes para Automação Industrial. Universidade Federal do Rio Grande do Norte.

Souza, Alessandro J. (2005), Sistema de gerência de informação de processos industriais via web, Dissertação de mestrado, Universidade Federal do Rio Grande do Norte.

Referências Bibliográficas

Elipse, Software (2007), Elipse scada - hmi/scada software - manual do usuário. Manual do Usuário.

Fieldbus, Foundation (20 05), Manual de instruções dos blocos funcionais *Foundation Fieldbus*.

Filho, Constantino Seixas (2000), A automação nos anos 2000 - uma análise das novas fronteiras da automação, Relatório técnico, ATAN Sistemas de Automação.

LAMP (2003), Manual teórico - lamp - laboratório de avaliação de medição em petróleo. REde 10/01.

Lima, Fábio S. (2004), Estratégia de escalonamento de controladores pid baseado em regras fuzzy para redes industriais foundation fieldbus usando blocos padrões, Dissertação de mestrado, Universidade Federal do Rio Grande do Norte.

Maitelli, André Laurindo (2003), Controladores lógicos programáveis. Notas de aula da disciplina "Controladores Lógicos Programáveis".

Neto, Alvaro Pereira (2003), Postgresql, técnicas avançadas versão open source 7.x.

Neves, Denise Lemes Fernandes (2002), Postgresql, conceitos e aplicações.

Silva, Ana Paula G. e Marcelo Salvador (2005), O que são sistemas supervisórios? http://www.elipse.com.br/. Visitada em Novembro de 2007.

Silva, Diego Rodrigo Cabral (2005), Redes neurais artificiais no ambiente de redes industriais *Foundation Fieldbus* usando blocos funcionais padrões, Dissertação de mestrado, Universidade Federal do Rio Grande do Norte.

Silveira, Leonardo e Weldson Q. Lima (2003), Um breve histórico conceitual da automação industrial e redes para automação industrial. Redes para Automação Industrial. Universidade Federal do Rio Grande do Norte.

Souza, Alessandro J. (2005), Sistema de gerência de informação de processos industriais via web, Dissertação de mestrado, Universidade Federal do Rio Grande do Norte.

ANEXO II

HISTÓRICO ESCOLAR

Histórico Escolar - Emitido em: 18/02/2008 às 15:19h

Dados Pessoais

Nome: **JEFERSON RIBEIRO DOS SANTOS JUNIOR** Matrícula **200320017**

Data de Nascimento: **15/10/1984** Local de Nascimento: **RIO GRANDE DO NORTE**

Nome do Pai: **JEFERSON RIBEIRO DOS SANTOS**

Nome da Mãe: **VALQUIRIA APARECIDA DOS SANTOS**

Endereço: **RUA MANOEL DE PININHA, 145** Bairro: **PONTA NEGRA**

Município: **NATAL** UF: **RN**

Dados do Curso

Curso: **ENGENHARIA DE COMPUTACAO - NATAL - PRESENCIAL - FORMAÇÃO - AUTOMACAO INDUSTRIAL - MTN**

Currículo: **02** Status: **ATIVO - GRADUANDO** IRA: **7,8188**

Reconhecimento do Curso: **PORTARIA Nº 1.515**

Data do Decreto: **16/07/2001** D.O.U.: **18/07/2001**

Ano/Período Letivo Inicial: **2003.2**

Forma de Ingresso: **VESTIBULAR**

Período Letivo Atual: **10** Prazo para Conclusão: **20121**

Trancamentos: **Nenhum** Perfil Inicial: **0**

Prorrogações: **0 períodos letivos**

Ano/Período Letivo de Saída: Participou do ENADE:

Tipo Saída: Data da Colação de Grau:

Trabalho de Conclusão de Curso:

Componentes Curriculares Cursadas/Cursando

Ano/Per			Componente Curricular	CH	Turma	Freq %	Nota	Situação
2003.2		DIM0322	INTRODUCAO A ENGENHARIA DE SOFTWARE	60	01	100.0	8.8	APROVADO
2003.2		DIM0323	MATEMATICA DISCRETA PARA COMPUTACAO	60	01	100.0	8.2	APROVADO
2003.2		DIM0324	CONCEITOS E TECNICAS DE PROGRAMACAO	60	01	100.0	9.8	APROVADO
2003.2		DIM0325	LABORATORIO DE CONCEITOS E TECNICAS DE PROGRAMACAO	45	02	100.0	9.5	APROVADO
2003.2		ELE0425	CIRCUITOS LOGICOS COMBINACIONAIS	60	01	95.0	8.4	APROVADO
2003.2		ELE0426	LABORATORIO DE CIRCUITOS LOGICOS COMBINACIONAIS	45	01	93.33	8.9	APROVADO
2003.2		MAT0228	CALCULO DIFERENCIAL E INTEGRAL I	90	01	100.0	7.7	APROVADO
2004.1		DIM0326	ALGORITMOS E ESTRUTURAS DE DADOS I	60	01	100.0	7.5	APROVADO
2004.1		DIM0327	LABORATORIO DE ALGORITMOS E ESTRUTURAS DE DADOS I	45	02	100.0	9.6	APROVADO
2004.1		ELE0427	CIRCUITOS LOGICOS SEQUENCIAIS	60	01	100.0	9.1	APROVADO
2004.1		ELE0428	LABORATORIO DE CIRCUITOS LOGICOS SEQUENCIAIS	45	01	100.0	9.0	APROVADO
2004.1		MAT0005	CALCULO DIFERENCIAL E INTEGRAL II	90	01	100.0	7.5	APROVADO
2004.1		MAT0230	GEOMETRIA ANALITICA E ALGEBRA LINEAR	90	01	96.66	7.0	APROVADO
2004.2		DCA0404	ARQUITETURA DE COMPUTADORES	60	01	100.0	7.3	APROVADO
2004.2		DCA0429	ANALISE DE SISTEMAS LINEARES	90	01	95.55	5.2	APROVADO
2004.2		DCA0430	CONTEXTO SOCIAL E PROFISSIONAL DA ENGENHARIA	30	01	100.0	6.3	APROVADO
2004.2		DIM0050	LOGICA APLICADA A COMPUTACAO	60	01	100.0	8.7	APROVADO
2004.2		DIM0328	ALGORITMOS E ESTRUTURAS DE DADOS II	60	01	100.0	5.1	APROVADO
2004.2		DIM0329	LABORATORIO DE ALGORITMOS E ESTRUTURAS DE DADOS II	45	02	100.0	9.0	APROVADO
2004.2		FIS0317	ELEMENTOS DE ELETRICIDADE E MAGNETISMO	60	01	80.0	3.1	REPROVADO
2005.1		DCA0451	COMPUTACAO NUMERICA	60	01	93.33	7.3	APROVADO
2005.1	e	DIM0049	TEORIA DA COMPUTACAO	60	01	93.33	7.2	APROVADO
2005.1		DIM0056	SOFTWARE BASICO	60	01	96.66	9.2	APROVADO
2005.1		EST0322	ESTATISTICA APLICADA A INFORMATICA	60	01	96.66	9.1	APROVADO
2005.1		FIS0317	ELEMENTOS DE ELETRICIDADE E MAGNETISMO	60	01	90.0	7.8	APROVADO
2005.1		MEC0404	MECANICA DOS SOLIDOS	90	01	97.77	7.0	APROVADO
2005.2		DCA0403	SISTEMAS DE TRANSMISSAO DE DADOS	60	01	100.0	8.2	APROVADO
2005.2		DCA0431	TEORIA DE CIRCUITOS	60	01	100.0	7.1	APROVADO
2005.2		DIM0338	SISTEMAS OPERACIONAIS	90	01	91.11	9.0	APROVADO
2005.2		DIM0339	COMPILADORES	90	01	100.0	4.5	REPROVADO

Histórico Escolar - Emitido em: 18/02/2008 às 15:19h

Nome: **JEFERSON RIBEIRO DOS SANTOS JUNIOR** Matrícula: **200320017**

Componentes Curriculares Cursadas/Cursando

Ano/Per			Componente Curricular	CH	Turma	Freq %	Nota	Situação
2005.2		DIM0340	FORMACAO HUMANISTICA EM COMPUTACAO	30	01	83.33	8.5	APROVADO
2005.2		FIS0316	FISICA EXPERIMENTAL II	45	11	93.33	8.2	APROVADO
2005.2		FIS0318	ELEMENTOS DE OTICA E ONDAS	60	01	100.0	7.3	APROVADO
2006.1	#	CON0002	CONTABILIDADE APLICADA A ADMINISTRACAO	60	02	100.0	6.5	APROVADO
2006.1		DCA0409	PROGRAMACAO EM TEMPO REAL	60	01	96.66	10.0	APROVADO
2006.1		DCA0433	MODELAGEM E ANALISE DE SISTEMAS DINAMICOS	60	01	93.33	6.5	APROVADO
2006.1		DCA0450	REDES DE COMPUTADORES	60	01	100.0	7.0	APROVADO
2006.1		DCA0900	INTELIGENCIA ARTIFICIAL APLICADA	60	01	100.0	7.5	APROVADO
2006.1		DIM0342	ALGORITMOS E ESTRUTURAS DE DADOS III	60	01	83.33	5.0	APROVADO
2006.1		ELE0401	ELETRONICA BASICA	60	01	96.66	5.9	APROVADO
2006.1		ELE0434	LABORATORIO DE ELETRONICA BASICA	30	02	100.0	8.4	APROVADO
2006.1	#	EST0220	ESTATISTICA APLICADA A ADMINISTRACAO I	60	02	100.0	9.3	APROVADO
2006.2		DCA0435	COMPUTACAO GRAFICA	60	01	93.33	8.0	APROVADO
2006.2		DCA0436	SISTEMAS DE CONTROLE	60	01	93.33	6.6	APROVADO
2006.2		DCA0437	LABORATORIO DE SISTEMAS DE CONTROLE	30	02	100.0	9.1	APROVADO
2006.2	*	DCA0444	PROJETO DE SISTEMAS MICROCONTROLADOS	60	01	90.0	8.9	APROVADO
2006.2	*	DIM0095	TOPICOS ESPECIAIS EM COMPUTACAO VI	60	01	100.0	5.6	APROVADO
2006.2	*	DIM0096	TOPICOS ESPECIAIS EM COMPUTACAO VII	60	01	100.0	7.4	APROVADO
2006.2		DIM0339	COMPILADORES	90	01	100.0	7.9	APROVADO
2007.1	*	DCA0406	REDES DE COMPUTADORES II	90	01	100.0	8.9	APROVADO
2007.1	*	DCA0407	INSTRUMENTACAO PARA CONTROLE E AUTOMACAO	60	01	100.0	9.4	APROVADO
2007.1	*	DCA0420	LINGUAGEM DE DESCRICAO DE HARDWARE	60	01	96.66	9.6	APROVADO
2007.1	*	DCA0449	TOPICOS ESPECIAIS EM REDES DE COMPUTADORES	60	01	93.33	7.0	APROVADO
2007.1	*	DCA0453	PROCESSAMENTO DIGITAL DE SINAIS	60	01	100.0	7.7	APROVADO
2007.1	*	DCA0454	REDES NEURAIS ARTIFICIAIS	60	01	100.0	8.0	APROVADO
2007.1		DIM0345	EMPREENDEDORISMO	60	01	98.33	8.9	APROVADO
2007.2	#	DCA0401	REDES DE SENSORES SEM FIO	60	01	--	--	TRANCADO
2007.2	*	DCA0443	CONTROLADORES LOGICOS PROGRAMAVEIS	60	01	100.0	9.3	APROVADO
2007.2	@	DCA0990	ESTAGIO SUPERVISIONADO	165	--	100.0	10.0	APROVADO
2007.2	*	DEF0650	ATIVIDADE FISICA, SAUDE E QUALIDADE DE VIDA	60	06	90.0	7.6	APROVADO
2007.2	*	DEQ0376	INTRODUCAO A ENGENHARIA DE PETROLEO	60	01	100.0	8.9	APROVADO
2007.2	&	MEC0015	METROLOGIA	60	01	93.33	7.3	APROVADO

Legenda:

* Comp. Optativo	e Comp. Equivalente a Obrig.	& Comp. Equivalente a Optativo	# Comp. Eletivo	@ Ativ. Obrigatória	§ Ativ. Optativa

	Obrigatórias			Complementares	Total	
	Comp. Curricular		Atividade	Comp.Curricular/Atividade		
	CR	CH	CH	CH	CR	CH
Exigido	171	2565	165	900	171	3630
Integralizado	171	2565	165	975	171	3705
Pendente	0	0	0	0	0	0

Componentes Curriculares Obrigatórios Pendentes: 0

Não há componentes curriculares obrigatórios pendentes

Equivalências:

Cumpriu DIM0337 - COMPUTABILIDADE (45h) através de DIM0049 - TEORIA DA COMPUTACAO (60h)

Atenção, agora o histórico possui uma verificação automática de autenticidade e consistência, sendo portanto dispensável a assinatura do coordenador ou do DAE. Favor, ler instruções no rodapé.

ANEXO III

PUBLICAÇÕES

“Sem Publicações”